

# 30th Conference on Computational Complexity

CCC'15, June 17-19, 2015, Portland, Oregon, USA

Edited by

David Zuckerman



*Editor*

David Zuckerman  
Department of Computer Science  
University of Texas at Austin  
2317 Speedway, Stop D9500  
Austin, Texas 78712  
USA  
diz@cs.utexas.edu

*ACM Classification 1998*  
F. Theory of Computation

**ISBN 978-3-939897-81-1**

*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/978-3-939897-81-1>.

*Publication date*

June, 2015

*Bibliographic information published by the Deutsche Nationalbibliothek*

The Deutsche Nationalbibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data are available from the Internet at <http://dnb.d-nb.de>.

*License*

This work is licensed under a Creative Commons Attribution 3.0 Unported license (CC-BY 3.0): <http://creativecommons.org/licenses/by/3.0/legalcode>.



In brief, this license authorizes each and everybody to share (to copy, distribute and transmit) the work under the following conditions, without impairing or restricting the authors' moral rights:

- Attribution: The work must be attributed to its authors.

The copyright is retained by the corresponding authors.

Digital Object Identifier: 10.4230/LIPIcs.CCC.2015.i

**ISBN 978-3-939897-81-1**

**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.

### *Editorial Board*

- Susanne Albers (TU München)
- Chris Hankin (Imperial College London)
- Deepak Kapur (University of New Mexico)
- Michael Mitzenmacher (Harvard University)
- Madhavan Mukund (Chennai Mathematical Institute)
- Catuscia Palamidessi (INRIA)
- Wolfgang Thomas (*Chair*, RWTH Aachen)
- Pascal Weil (CNRS and University Bordeaux)
- Reinhard Wilhelm (Saarland University)

**ISSN 1868-8969**

**<http://www.dagstuhl.de/lipics>**





## ■ Contents

Preface	
<i>David Zuckerman</i> .....	vii
Awards	
.....	ix
Conference Organization	
.....	xi
External Reviewers	
.....	xiii

### Contributed Papers

Strong Locally Testable Codes with Relaxed Local Decoders	
<i>Oded Goldreich, Tom Gur, and Ilan Komargodski</i> .....	1
An Entropy Sunset Inequality and Polynomially Fast Convergence to Shannon Capacity Over All Alphabets	
<i>Venkatesan Guruswami and Ameya Velingker</i> .....	42
The List-Decoding Size of Fourier-Sparse Boolean Functions	
<i>Ishay Haviv and Oded Regev</i> .....	58
Nonclassical Polynomials as a Barrier to Polynomial Lower Bounds	
<i>Abhishek Bhowmick and Shachar Lovett</i> .....	72
Simplified Lower Bounds on the Multiparty Communication Complexity of Disjointness	
<i>Anup Rao and Amir Yehudayoff</i> .....	88
How to Compress Asymmetric Communication	
<i>Sivaramakrishnan Natarajan Ramamoorthy and Anup Rao</i> .....	102
Majority is Incompressible by $AC^0[p]$ Circuits	
<i>Igor Carboni Oliveira and Rahul Santhanam</i> .....	124
Lower Bounds for Depth Three Arithmetic Circuits with Small Bottom Fanin	
<i>Neeraj Kayal and Chandan Saha</i> .....	158
A Depth-Five Lower Bound for Iterated Matrix Multiplication	
<i>Suman K. Bera and Amit Chakrabarti</i> .....	183
Factors of Low Individual Degree Polynomials	
<i>Rafael Oliveira</i> .....	198
Verifiable Stream Computation and Arthur-Merlin Communication	
<i>Amit Chakrabarti, Graham Cormode, Andrew McGregor, Justin Thaler, and Suresh Venkatasubramanian</i> .....	217
Identifying an Honest $EXP^{NP}$ Oracle Among Many	
<i>Shuichi Hirahara</i> .....	244

30th Conference on Computational Complexity (CCC'15).

Editor: David Zuckerman



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

Adaptivity Helps for Testing Juntas <i>Rocco A. Servedio, Li-Yang Tan, and John Wright</i> .....	264
A Characterization of Hard-to-cover CSPs <i>Amey Bhangale, Prahladh Harsha, and Girish Varma</i> .....	280
Subexponential Size Hitting Sets for Bounded Depth Multilinear Formulas <i>Rafael Oliveira, Amir Shpilka, and Ben Lee Volk</i> .....	304
Deterministic Identity Testing for Sum of Read-once Oblivious Arithmetic Branching Programs <i>Rohit Gurjar, Arpita Korwar, Nitin Saxena, and Thomas Thierauf</i> .....	323
Kolmogorov Width of Discrete Linear Spaces: an Approach to Matrix Rigidity <i>Alex Samorodnitsky, Ilya Shkredov, and Sergey Yekhanin</i> .....	347
On the (Non) NP-Hardness of Computing Circuit Complexity <i>Cody D. Murray and R. Ryan Williams</i> .....	365
Circuits with Medium Fan-In <i>Pavel Hrubeš and Anup Rao</i> .....	381
Correlation Bounds Against Monotone $NC^1$ <i>Benjamin Rossman</i> .....	392
Non-Commutative Formulas and Frege Lower Bounds: a New Characterization of Propositional Proofs <i>Fu Li, Iddo Tzameret, and Zhengyu Wang</i> .....	412
The Space Complexity of Cutting Planes Refutations <i>Nicola Galesi, Pavel Pudlák, and Neil Thapen</i> .....	433
Tight Size-Degree Bounds for Sums-of-Squares Proofs <i>Massimo Lauria and Jakob Nordström</i> .....	448
A Generalized Method for Proving Polynomial Calculus Degree Lower Bounds <i>Mladen Mikša and Jakob Nordström</i> .....	467
Generalized Quantum Arthur-Merlin Games <i>Hirota Kobayashi, François Le Gall, and Harumichi Nishimura</i> .....	488
Parallel Repetition for Entangled $k$ -player Games via Fast Quantum Search <i>Kai-Min Chung, Xiaodi Wu, and Henry Yuen</i> .....	512
Upper Bounds on Quantum Query Complexity Inspired by the Elitzur-Vaidman Bomb Tester <i>Cedric Yen-Yu Lin and Han-Hsuan Lin</i> .....	537
A Polylogarithmic PRG for Degree 2 Threshold Functions in the Gaussian Setting <i>Daniel M. Kane</i> .....	567
Incompressible Functions, Relative-Error Extractors, and the Power of Nondeterministic Reductions (Extended Abstract) <i>Benny Applebaum, Sergei Artemenko, Ronen Shaltiel, and Guang Yang</i> .....	582
On Randomness Extraction in $AC^0$ <i>Oded Goldreich, Emanuele Viola, and Avi Wigderson</i> .....	601

## ■ Preface

The papers in this volume were accepted for presentation at the 30th Computational Complexity Conference (CCC'15), held June 17–19, 2015 in Portland, Oregon as part of the ACM Federated Computing Research Conference (FCRC'15). The conference is organized by the Computational Complexity Foundation in cooperation with the European Association for Theoretical Computer Science (EATCS) and the ACM Special Interest Group on Algorithms and Computation Theory (SIGACT). CCC'15 is sponsored by Microsoft Research and is supported by the Institute for Quantum Computing (IQC).

The call for papers sought original research papers in all areas of computational complexity theory. Of the 110 submissions the program committee selected 30 for presentation at the conference.

The program committee would like to thank everyone involved in the conference, including all those who submitted papers for consideration as well as the reviewers (listed separately) for their scientific contributions; the board of trustees of the Computational Complexity Foundation and especially its president Dieter van Melkebeek for extensive advice and assistance; Jacobo Toran for a variety of assistance; Mike Saks for sharing his knowledge as 2014 PC chair; and Marc Herbstritt for coordinating the production of these proceedings.

David Zuckerman  
Program Committee Chair





## ■ Awards

The program committee of the 30th Computational Complexity Conference is happy to present the 2015 Best Paper Award to Benjamin Rossman for his paper

“Correlation Bounds Against Monotone  $NC^1$ ,”

and the 2015 Best Student Paper Award to Rafael Oliveira for his paper

“Factors of Low Individual Degree Polynomials.”

Funding for the best student paper award is provided by the European Association for Theoretical Computer Science (EATCS).





## ■ Conference Organization

### **Program Committee**

Andris Ambainis, University of Latvia  
Andrej Bogdanov, Chinese University of Hong Kong  
Andrew Drucker, University of Edinburgh  
Zeev Dvir, Princeton University  
Parikshit Gopalan, Microsoft  
Johan Håstad, KTH, Stockholm  
Russell Impagliazzo, University of California, San Diego  
Prasad Raghavendra, University of California, Berkeley  
Ran Raz, Weizmann Institute & IAS  
Shubhangi Saraf, Rutgers University  
David Zuckerman (chair), University of Texas at Austin

### **FCRC Liaison**

Dieter van Melkebeek, University of Wisconsin-Madison

### **Board of Trustees**

Eric Allender (Treasurer), Rutgers University  
Venkatesan Guruswami, Carnegie Mellon University  
Jeff Kinne (Secretary), Indiana State University  
Dieter van Melkebeek (President), University of Wisconsin-Madison  
Madhu Sudan, Microsoft Research New England  
Jacob Toran, University of Ulm  
Osamu Watanabe, Tokyo Institute of Technology







## ■ External Reviewers

Alex Arkhipov	Per Austrin	Raef Bassily
Salman Beigi	Aleksandrs Belovs	Arnab Bhattacharyya
Abhishek Bhowmick	Eric Blais	Fernando Brandao
Vladimir Braverman	Jop Briet	Joshua Brody
Christina Brzuska	Andre Chailloux	Sourav Chakraborty
Siu On Chan	Xi Chen	Kai-Min Chung
Julia Chuzhoy	Constantinos Daskalakis	Anindya De
Javier Esparza	Will Evans	Omar Fawzi
Yuval Filmus	Michael Forbes	Ariel Gabizon
Nicola Galesi	Ankit Garg	Sivakanth Gopi
Alan Gou	Joao Gouveia	Siyao Guo
Aram Harrow	Prahladh Harsha	Sangxia Huang
Christian Ikenmeyer	Janis Iraids	Erich Kaltofen
Michael Kapralov	Tali Kaufman	Akinori Kawachi
Subhash Khot	Hartmut Klauck	Gillat Kol
Swastik Kopparty	Ravi Kumar	Massimo Lauria
Chin Ho Lee	Troy Lee	Nikos Leonardos
Satyanarayana Lokam	Shachar Lovett	Laura Mancinska
Raghu Meka	Mladen Miksa	Tomoyuki Morimae
Dana Moshkovitz	Daniel Nagaj	Chandra Nair
Hariharan Narayanan	Jakob Nordstrom	Ryan O'Donnell
Toni Pitassi	Sebastian Pokutta	Igor Razgon
Oded Regev	Silas Richelson	Aviad Rubinstein
Rahul Santhanam	Giannicola Scarpa	Rocco Servedio
Alexander Sherstov	Seung Woo Shin	Srikanth Srinivasan
Piyush Srivastava	David Steurer	Howard Straubing
Ido Tal	Li-Yang Tan	Luca Trevisan
Hing Yin Tsang	Salil Vadhan	Greg Valiant
Dieter van Melkebeek	Thomas Vidick	Jevgenijs Vihrovs
Marc Vinyals	Emanuele Viola	Ben Lee Volk
Carol Wang	Osamu Watanabe	John Watrous
Thomas Watson	Omri Weinstein	Ryan Williams
Karl Wimmer	John Wright	Sergey Yekhanin
Ke Yi	Shengyu Zhang	





# Strong Locally Testable Codes with Relaxed Local Decoders\*

Oded Goldreich, Tom Gur, and Ilan Komargodski

Weizmann Institute of Science  
Rehovot, Israel

{oded.goldreich,tom.gur,ilan.komargodski}@weizmann.ac.il

---

## Abstract

---

Locally testable codes (LTCs) are *error-correcting codes* that admit very efficient codeword tests. An LTC is said to be **strong** if it has a *proximity-oblivious* tester; that is, a tester that makes only a *constant number* of queries and reject non-codewords with probability that depends solely on their distance from the code.

Locally decodable codes (LDCs) are complimentary to LTCs. While the latter allow for highly efficient rejection of strings that are far from being codewords, LDCs allow for highly efficient recovery of individual bits of the information that is encoded in strings that are close to being codewords.

Constructions of strong-LTCs with nearly-linear length are known, but the existence of a constant-query LDC with *polynomial* length is a major open problem. In an attempt to bypass this barrier, Ben-Sasson *et al.* (SICOMP 2006) introduced a natural relaxation of local decodability, called relaxed-LDCs. This notion requires local recovery of nearly all individual information-bits, yet allows for recovery-failure (but not error) on the rest. Ben-Sasson *et al.* constructed a constant-query relaxed-LDC with nearly-linear length (i.e., length  $k^{1+\alpha}$  for an arbitrarily small constant  $\alpha > 0$ , where  $k$  is the dimension of the code).

This work focuses on obtaining strong testability and relaxed decodability *simultaneously*. We construct a family of binary linear codes of nearly-linear length that are both strong-LTCs (with one-sided error) and constant-query relaxed-LDCs. This improves upon the previously known constructions, which either obtain weak LTCs or require polynomial length.

Our construction heavily relies on *tensor codes* and PCPs. In particular, we provide *strong canonical* PCPs of *proximity* for membership in any linear code with constant rate and relative distance. Loosely speaking, these are PCPs of *proximity* wherein the verifier is proximity oblivious (similarly to strong-LTCs) and every valid statement has a unique *canonical* proof. Furthermore, the verifier is required to reject non-canonical proofs (even for valid statements).

As an application, we improve the best known separation result between the complexity of *decision* and *verification* in the setting of property testing.

**1998 ACM Subject Classification** F.1.3 [Computation by Abstract Devices] Complexity Measures and Classes

**Keywords and phrases** Locally Testable Codes, Locally Decodable Codes, PCPs of Proximity

**Digital Object Identifier** 10.4230/LIPIcs.CCC.2015.1

---

\* This research was partially supported by the Minerva Foundation with funds from the Federal German Ministry for Education and Research, and by a grant from the I-CORE Program of the Planning and Budgeting Committee, the Israel Science Foundation and the Citi Foundation.



© Oded Goldreich, Tom Gur, and Ilan Komargodski;  
licensed under Creative Commons License CC-BY

30th Conference on Computational Complexity (CCC'15).

Editor: David Zuckerman; pp. 1–41



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

## 1 Introduction

*Locally testable codes* (LTCs) are error-correcting codes that can be tested very efficiently. Specifically, a code is said to be an LTC if there exists a probabilistic algorithm, called a **tester**, that is given a *proximity parameter*  $\varepsilon > 0$  and oracle access to an input string (an alleged codeword), makes a small number (e.g.,  $\text{poly}(1/\varepsilon)$ ) of queries to the input and is required to accept valid codewords, and reject with high probability input strings that are  $\varepsilon$ -far from being a codeword (i.e., reject strings that disagree with any codeword on  $\varepsilon$  fraction of the bits). The systematic study of LTCs was initiated by Goldreich and Sudan [13], though the notion was mentioned, in passing, a few years earlier by Friedl and Sudan [8] and Rubinfeld and Sudan [20].

A natural strengthening of the notion of locally testable codes (LTCs) is known as **strong-LTCs**. While LTCs (also referred to as **weak-LTCs**) allow for a different behavior of the tester for different values of the proximity parameter, **strong-LTCs** are required to satisfy a strong *uniformity* condition over all values of the proximity parameter. In more detail, the tester of a **strong-LTC** does *not* get a proximity parameter as an input, and is instead required to make only a *constant* number of queries and reject non-codewords with probability that is related to their distance from the code. See [13, 10] for a discussion on both types of local testability. We note that from a property testing point of view, **strong-LTCs** can be thought of as codes that can be tested by a *proximity-oblivious tester* (see [12]).

The two most fundamental parameters of error-correcting codes (and **strong-LTCs** in particular) are the *distance* and the codeword *length*. Throughout this work we will only consider codes with constant relative distance, and so our main parameter of interest is the length, which measures the amount of redundancy of information in each codeword. By this criterion, constructing a **strong-LTC** with linear length (and constant relative distance) is the holy grail of designing efficient locally testable codes. Although recently some progress was made towards showing the impossibility of such linear length LTCs [5, 3], there are known constructions of **strong-LTCs** with relatively good parameters: Goldreich and Sudan [13] constructed a **strong-LTC** with constant relative distance and nearly-linear length, where throughout this paper a code of dimension  $k$  is said to have *nearly-linear length* if its codewords are of length  $k^{1+\alpha}$  for an arbitrarily small constant  $\alpha > 0$ . Furthermore, recently Viderman [23] constructed a **strong-LTC** with constant relative distance and quasilinear length (i.e., length  $k \cdot \text{polylog}k$ ).

Another natural local property of codes is *local decodability*. A code is said to be a **locally decodable code** (LDC) if it allows for a highly efficient recovery of any individual bit of the message encoded in a *somewhat corrupted* codeword. That is, there exists a probabilistic algorithm, called a **decoder**, that is given a location  $i$  and oracle access to an input string  $w$  that is promised to be sufficiently close to a codeword. The decoder is allowed to make a small (usually constant) number of queries to the input  $w$  and is required to decode the  $i^{\text{th}}$  bit of the information that corresponds to the codeword that  $w$  is closest to. Following the work of Katz and Trevisan [17] that formally defined the notion of LDCs, these codes received much attention and found numerous applications (see e.g., [21, 25] and references therein). They are also related to *private information retrieval* protocols [4] (see [9] for a survey).

Despite much attention that LDCs received in recent years, the best known LDCs are of super-polynomial length (cf. [7], building on [24]). While the best known lower bound (cf. [17]) only shows that any  $q$ -query LDC must be of length  $\Omega\left(k^{1+\frac{1}{q-1}}\right)$  (where  $k$  is the dimension of the code), the existence of a constant-query LDC with *polynomial* length remains a major open problem.

In an attempt to bypass this barrier, Ben-Sasson *et al.* [1] introduced a natural relaxation of the notion of local decodability, known as relaxed-LDCs. This relaxation requires local recovery of most (or nearly all) individual information-bits, yet allows for recovery-failure (but not error) on the rest. Specifically, a code is said to be a relaxed-LDC if there exists an algorithm, called a (relaxed) decoder, that has oracle access to an input string that is promised to be sufficiently close to a codeword. Similarly to LDCs, the decoder is allowed to make few queries to the input in attempt to decode a given location in the message. However, unlike LDCs, the relaxed decoder is allowed to output an abort symbol on a small fraction of the locations, which indicates that the decoder detected a corruption in the codeword and is unable to decode this specific information-bit. Note that the decoder must still avoid errors (with high probability).

Throughout this work, unless explicitly stated otherwise, when we say that a code is a relaxed-LDC, we actually mean that it is a relaxed-LDC with *constant* query complexity.

Ben-Sasson *et al.* [1] constructed a relaxed-LDC with nearly-linear length. More generally, they showed that for every constant  $\alpha > 0$  there exists a relaxed-LDC (with constant relative distance) that maps  $k$ -bit messages to  $k^{1+\alpha}$ -bit codewords and has query complexity  $O(1/\alpha^2)$ . While these relaxed-LDCs are dramatically shorter than any known LDC, they do not break the currently known lower bound on LDCs (cf. [17]), and hence it is still an open question whether relaxed-LDC are a strict relaxation of LDCs.

## 1.1 Obtaining Local Testability and Decodability Simultaneously

In this work, we are interested in short codes that are both (strongly) locally testable and (relaxed) locally decodable.<sup>1</sup> The motivation behind such codes is very natural, as the notion of local decodability is complimentary to the notion of local testability: The success of the decoding procedure of a locally decodable code is pending on the promise that the input is sufficiently close to a valid codeword. If the locally decodable code is also locally testable, then this promise can be verified by the testing procedure. However, recall that there are no known constant-query LDCs with even polynomial length, let alone such that are also locally testable. Hence, we focus on relaxed-LDCs.<sup>2</sup>

There are a couple of known constructions of codes that are both locally testable and relaxed decodable (with constant query complexity). Ben-Sasson *et al.* [1] observed that their relaxed-LDC can be modified to also be a weak-LTC (i.e., an LTC that is not strong), while keeping its length nearly-linear. However, the local testability of their code is inherently *weak* (see Section 1.3 for details). In a recent development, Gur and Rothblum [15] constructed a relaxed-LDC that is also a strong-LTC, albeit with *polynomial length*.

In this paper, we improve upon the aforementioned results of [1] and [15], achieving the best of both worlds. That is, we construct a code that is both a strong-LTC and a relaxed-LDC with *nearly-linear* length.

► **Theorem 1.1** (informal). *There exists a binary linear code that is a relaxed-LDC and a (one-sided error) strong-LTC with constant relative distance and nearly-linear length.*

<sup>1</sup> Note that although the notion of local testability and decodability are related, LTCs do not imply LDCs (i.e., there are LTCs that are not LDCs) and vice-versa. (See [18].)

<sup>2</sup> A different possible approach to solve this problem is to settle for codes with *long length*. Indeed, there are codes with *exponential* length that are both (constant-query) LDCs and LTCs, e.g., the Hadamard code. Another approach to solve this problem is to settle for codes with *large query complexity*. In a recent work, Guo, Kopparty, and Sudan [14] constructed very short length codes that are both locally testable and locally decodable, albeit with large (yet needless to say, sub-linear) query complexity.

A formal statement of Theorem 1.1 is given in Section 3. We remark that we actually prove a slightly stronger claim; namely, that any good linear code can be augmented (by appending additional bits to each codeword) into a code that is both a relaxed-LDC and a strong-LTC, at the cost of increasing the codeword length from linear to nearly-linear.

### On Invoking Testers Prior to Decoders

Recall that for a code that is both locally testable and decodable, the promise (that the input is close to a codeword) required by the decoder can be eliminated by invoking the tester first. However, doing so can potentially hamper the decodability, since the tester is allowed to reject codewords that are only slightly corrupted. Fortunately, our tester is smooth (i.e., it queries each of the  $n$  bits of a codeword with probability  $\Theta(1/n)$ ), and thus invoking the strong-tester a carefully chosen number of times (rejecting if one of the invocations rejected) will result in a tolerant tester (see [16, 19]). Such a tester will reject inputs that do not satisfy the promise of the decoder, yet still accept slightly-corrupted codewords (with high probability).

## 1.2 Strong Canonical PCPs of Proximity

The notion of PCPs of proximity plays a major role in many constructions of LTCs and relaxed-LDCs, as well as in our own. Loosely speaking, PCPs of proximity (PCPPs) are a variant of PCP proof systems, which can be thought of as the PCP analogue of *property testing*. Recall that a standard PCP is given explicit access to a statement (i.e., an input that is supposedly in some NP language) and oracle access to a proof (i.e., a “probabilistically checkable” NP witness). The PCP verifier is required to probabilistically verify whether the (explicitly given) statement is correct, by making few queries to the alleged proof. In contrast, a PCPP is given oracle access to a statement and to a proof, and is only allowed to make a small number of queries to both the statement and the proof. Since a PCPP verifier only sees a small part of the statement (typically, only a constant number of bits), it cannot be expected to verify the statement precisely. Instead, it is required only to accept correct statements and reject statements that are far from being correct (i.e., far in Hamming distance from any valid statement).

PCPs of proximity were first studied by Ben-Sasson *et al.* [1] and by Dinur and Reingold [6] (wherein they are called **assignment testers**). The main parameters of interest in a PCPP system for some language  $L$  are its *query complexity* (i.e., the total number of queries to the input and to the proof that the PCPP verifier makes in order to determine membership in  $L$ ) and its *proof length*, which can be thought as measuring the amount of redundancy of information in the proof. Ben-Sasson *et al.* [1] showed a PCPP for any language in NP, with constant query complexity and nearly-linear length (in fact, the length is  $n^{1+o(1)}$ , where  $n$  is the length of the corresponding NP-witness).

As we have already noted, PCPPs have a central theoretical significance as the property testing analogue of PCP proof-systems. Moreover, PCPPs were shown to be useful in various applications, e.g., for PCP composition and alphabet reduction [1, 6], and for locally testable and locally decodable codes [1, 13, 15]. Further information regarding the latter application follows.

The notion of locally testable codes and PCPs of proximity are tightly connected. Not only that PCPPs (and PCPs in general) can be thought of as the computational analogue of the (combinatorial) notion of LTCs, but also any code can be made locally testable by using an adequate PCPP. Specifically, Ben-Sasson *et al.* [1] showed that any linear code can be

transformed to a (weak) LTC by appending each codeword with a PCPP proof that ascertains that the codeword is indeed properly encoded.<sup>3</sup> However, since there is no guarantee that every two different proofs for the same statement are far (in Hamming distance), in order to prevent deterioration of the distance of the code two additional steps are taken: Firstly, the appended PCPP proof should be uniquely determined per codeword (i.e., each codeword has a *canonical* proof), and secondly, each codeword is repeated many times so that the PCPP part constitutes only a small fraction of the total length.

The drawback of the foregoing approach is that it results in locally testable codes that are *inherently* weak (i.e., codes that do not allow for proximity-oblivious testing). To see this, note that PCPPs only guarantee that false assertions are rejected (with high probability), while true assertions can be accepted even if the proof is incorrect. Hence, corruptions in the PCPP part are not necessarily detectable and the *canonicity* of the PCPP proofs may not be verified, ruling out the possibility of a (strong) tester that is uniform over all possible values of the proximity parameter.<sup>4</sup> Moreover, when trying to build **strong-LTCs**, an additional problem that arises is that, by definition, PCPPs do not necessarily provide strong soundness, i.e., reject false proofs with probability that depends only on their distance from a correct proof.

Motivated by constructing *strong* locally testable codes, Goldreich and Sudan [13, Section 5.3] considered a natural strengthening of the notion of PCPPs, known as **strong canonical PCPs of proximity** (hereafter **scPCPP**), which addresses the aforementioned issues. Loosely speaking, **scPCPP** are PCPPs with *strong soundness* that are required to reject “wrong” proofs, even for correct statements. Moreover, they require that each correct statement will only have one acceptable proof. In more detail, **scPCPP** are PCPP with two additional requirements: (1) *canonicity*: for every true statement there exists a unique proof (called the **canonical proof**) that the verifier is required to accept, and any other proof (even for a correct statement) must be rejected, and (2) *strong soundness*: the **scPCPP** verifier is required to be proximity oblivious and reject any pair of statement and proof with probability that is related to its distance from a true statement and its corresponding canonical proof. A formal definition of **scPCPPs** can be found in Section 2.4.

Given a construction of adequate **scPCPPs**, the aforementioned strategy of appending each codeword with an efficient **scPCPP** (which ascertains membership in a code) will allow to transform any code to a **strong-LTC**. Unfortunately, unlike standard PCPPs, for which there are efficient constructions for any language in NP, there are no known constructions of general-purpose **scPCPPs**. Yet, Goldreich and Sudan constructed a mechanism, called *linear inner proof systems* (LIPS), which is closely related to some special-purpose **scPCPPs**. Loosely speaking, the LIPS mechanism allows to transform linear strong locally testable codes over a large alphabet into strong locally testable codes over a smaller alphabet (see [13, Section 5.2] for further details). By a highly non-trivial construction and usage of the LIPS mechanism, Goldreich and Sudan showed efficient constructions of **strong-LTCs**. Unfortunately, their constructions do not meet our needs. Nevertheless, building upon their techniques, we show *strong canonical PCPs of proximity* with polynomial length for any good linear code.

<sup>3</sup> Note that membership in any linear code is in P, and so, the efficient PCPP for NP of Ben-Sasson *et al.* [1] can handle these statements.

<sup>4</sup> In contrast, note that for *weak* LTCs this problem can be ignored by simply making the PCPPs themselves a sufficiently small part of the codewords. Recall that *weak* LTCs are allowed to only work for values of the proximity parameter that are sufficiently large to ensure that the concatenation of a corrupted codeword and its corresponding PCPP sequence will include (significant) corruption in the codeword part. Thus, there is no need to verify the canonicity or even validity of the PCPP proof. However, when we seek to achieve the stronger definition of LTCs (i.e., **strong-LTCs**), this problem becomes relevant (and cannot be ignored).

► **Theorem 1.2** (scPCPP for good codes – informal). *Let  $C$  be a linear code with constant relative distance and linear length. Then, there exists a scPCPP with polynomial proof length for membership in the set of all codewords of  $C$ .*

In fact, we actually prove a slightly stronger statement. Specifically, our scPCPPs satisfy two additional properties that will be useful for our main construction: The scPCPP proofs are linear (over  $\text{GF}(2)$ ), and the queries that the verifier makes are roughly uniform. We remark that not only that the scPCPPs in Theorem 1.2 are crucial to our construction (see Section 1.4 for details), we also view these scPCPPs as interesting on their own. A formal statement of Theorem 1.2 and its proof are presented in Section 6.

### 1.3 Previous Works and Techniques

In this subsection, we survey the previous works and techniques regarding relaxed-LDCs upon which we build. We start by recalling the construction of the (nearly) quadratic length relaxed-LDC of Ben-Sasson *et al.* [1, Section 4.2]. The core idea that underlies their construction is to partition each codeword into three parts: The first providing the distance property, the second allowing for “local decodability”, and the third ascertaining the consistency of the first two parts. The natural decoder for such a code will verify the consistency of the first two parts via the third part and decode according to the second part in case it detects no consistency error. Details follow.

Let  $C$  be any good linear code (i.e., a code with constant relative distance and linear length). Ben-Sasson *et al.* construct a new code  $C'$  whose codewords consist of three parts of equal length: (1) repetitions of a good codeword  $C(x)$  that encodes the message  $x$ ; (2) repetitions of the explicitly given message  $x$ ; and (3) PCPPs that ascertain the consistency of each individual bit in the message  $x$  (which is explicitly given in the second part) with the codeword  $C(x)$  (which is explicitly given in the first part). We remark that since the total length of the PCPPs is significantly longer than the statements they ascertain, the desired length proportions are obtained by the repetitions in the first two parts. Observe that the first part grants the new code  $C'$  good distance (although it may *not* be locally decodable), the second part allows for a highly efficient decoding of the message (at the cost of reducing the distance), and the third part is needed in order to guarantee that the first two parts refer to the same message. The (relaxed) decoder for  $C'$  will use the PCPPs in the third part in order to verify that the first part (the codeword  $C(x)$ ) is consistent with the bit we wish to decode in the second part (the message  $x$ ). If the PCPP verifier detects no error, the decoder returns the relevant bit in the second part; otherwise, it returns an abort symbol.

In order to implement the aforementioned relaxed-LDC, an adequate PCPP is needed; that is, an efficient PCPP for verifying the consistency of each individual bit in a message  $x$  with the codeword  $C(x)$ . We note that such statements are in P. Recall that Ben-Sasson *et al.* [1, Section 3] showed PCPPs with nearly-linear length for *any* language in NP. Hence, the consistency of each message bit with a codeword of  $C$  can be guaranteed by a PCPP of length that is nearly-linear in the length of  $C$ . Since  $C'$  is obtained by augmenting a good linear code  $C$  with a single PCPP proof per every message bit (claiming consistency between that bit and the codeword of  $C$ ), the length of  $C'$  is (nearly) quadratic (i.e., length  $k^{2+\alpha}$  for an arbitrarily small constant  $\alpha > 0$ , where  $k$  is the dimension of the code). We note that Ben-Sasson *et al.* showed that the length of  $C'$  can be improved to nearly-linear by, roughly speaking, partitioning the message into blocks of various lengths and decoding based on a chain of consistent blocks.<sup>5</sup>

<sup>5</sup> To obtain length  $k^{1.5+\alpha}$ , the message is partitioned into  $\sqrt{k}$  blocks, each of length  $\sqrt{k}$ . Then, the



Recall that *any* code can be transformed to a weak locally testable code by appending adequate PCPPs to it (See [1, Section 4.1]). Applying this transformation to the relaxed-LDC does not hamper the relaxed decodability of the code, and only increase its length by a moderate amount (since the PCPPs are of nearly-linear length); hence this transformation yields a (constant query) relaxed-LDC with nearly-linear length that is also a (weak) LTC. We stress that the aforementioned transformation yields local testability that is *inherently weak* due to the fact that it uses standard PCPPs. However, if the PCPPs in use were actually scPCPPs (of nearly-linear length), then the foregoing code would have been strongly testable.

In a recent work, Gur and Rothblum [15] constructed scPCPPs with polynomial length for the particular family of linear length statements that are needed for the [1] relaxed-LDC. By using these scPCPPs in the construction of [1], they obtained a relaxed-LDC that is also a strong-LTC, albeit with polynomial length (due to the length of their scPCPPs). While we conjecture it is feasible to construct nearly-linear length scPCPPs for P (which contains the family of statements we wish to have scPCPPs for) and even for unique-NP (also known as the class US),<sup>6</sup> we do not obtain such scPCPPs here. Instead, we take an alternative approach, which circumvents this challenge, as described in the next subsection.

## 1.4 Our Techniques

In this subsection, we present our main techniques and ideas for constructing a relaxed-LDC with nearly-linear length that is also a strong-LTC. Our starting point is the (*weakly* testable) relaxed-LDC construction of Ben-Sasson *et al.* [1]. However, we wish to replace the PCPPs that they use with scPCPPs, in order to achieve *strong* local testability.

Since we do not have general-purpose scPCPPs (let alone of near-linear length), we construct special-purpose scPCPPs that allow us to ascertain the particular statements we are interested in (see Theorem 1.2). It is crucial to note that the scPCPPs we are able to construct are with *polynomial* proof length (and not nearly-linear length, as we would have hoped). Recall that the statements that are needed for the construction of Ben-Sasson *et al.* (i.e., ascertaining the consistency of each bit of the message with the entire codeword for *decodability*, and ascertaining the validity of the codeword for *testability*) are linear in the length of the message. Therefore, applying our scPCPPs in a naive way (i.e., replacing the PCPPs in the construction of Ben-Sasson *et al.* with our scPCPPs) would yield codes with *polynomial* length, whereas we are aiming for nearly-linear length. Instead, we use an alternative approach.

The key idea is to provide scPCPPs that only refer to sufficiently short statements such that even with the polynomial blow-up of the scPCPP, the length of each proof would still be sub-linear. Specifically, instead of providing proofs for the validity of the entire codeword and the consistency of each message bit with the entire codeword (as in [1]), we provide proofs for the consistency of each message bit with “small” parts of the code and for the validity of

---

original message, as well as each of the smaller blocks is encoded by an error-correcting code. For each of the encoded smaller blocks, the following PCPPs are added: (1) a PCPP that ascertains the consistency of the encoded block with the encoded original-message; and (2) PCPPs that ascertains the consistency of each bit in the encoded block with the entire encoded block. Observe that the total encoding length decreased, since there are  $\sqrt{k}$  proofs of statements of length  $O(k)$  and  $k$  proofs of statements of length  $O(\sqrt{k})$ , thus, the total length is nearly-linear in  $k^{3/2}$ . By repeating this process, we can reduce the length of the code to  $k^{1+\alpha}$  for an arbitrarily small constant  $\alpha > 0$  (see [1, Section 4.2] for details).

<sup>6</sup> We note that the class unique-NP (i.e., the class of NP problems with unique witnesses) seems more likely to have scPCPPs than NP. This is because a language in NP may have many witnesses per instance, and it is not clear how to recognize the “canonical” NP-witness.

these small parts. If each part is sufficiently small (i.e., of length  $k^\alpha$  for an arbitrarily small constant  $\alpha > 0$ , where  $k$  is the length of the message), then we can still obtain a code with nearly-linear length, even when providing polynomial length proofs for all of the small parts.

The caveat, however, is that proving that each message bit is consistent with a small part (or *local view*) of a codeword does *not* necessarily imply that the message bit is consistent with the entire codeword. Similarly, partitioning a codeword into small parts and proving the validity of each part does not imply the validity of the entire codeword. Therefore, we need the base code (to which we append scPCPPs) to be highly structured so that, loosely speaking, the *local* consistency and validity we are able to ascertain can be used to enforce *global* consistency and validity. Concretely, the strategy we employ is using *tensor codes* and proving that this family of codes has features that allow us to overcome the aforementioned caveat. Details follow.

Given a linear code  $C : \{0, 1\}^k \rightarrow \{0, 1\}^n$ , the tensor code  $C \otimes C : \{0, 1\}^{k^2} \rightarrow \{0, 1\}^{n^2}$  consists of all  $n \times n$  matrices whose rows and columns are codewords of  $C$ . Similarly, the  $d$ -dimensional tensor code  $C^{\otimes d} = \underbrace{C \otimes C \otimes \cdots \otimes C}_{d \text{ times}} : \{0, 1\}^{k^d} \rightarrow \{0, 1\}^{n^d}$  is defined in the natural way. Namely,  $C^{\otimes d}$  consists of all  $\underbrace{n \times n \times \cdots \times n}_{d \text{ times}}$ -dimensional tensors such that each (axis-parallel) line in the tensor is a codeword of  $C$ .<sup>7</sup> (See Section 2.3 for the exact definitions.)

Towards obtaining relaxed local decodability, we show that tensor codes satisfy a feature, which we call *local propagation*, that allows us to verify *global* consistency statements (such as the ones that are used in the [1] relaxed-LDC) by verifying *local* consistency statements, which we can afford to prove with *polynomial* length scPCPPs; the *local propagation* feature of tensor codes is discussed in Section 4. Hence, we can ascertain that the value at each point in the tensor is consistent with the entire codeword by verifying the consistency of a constant number of randomly selected statements regarding small parts of the tensor (specifically, statements of consistency between the value at a point in the tensor and a line that passes through it). We remark that Theorem 1.2 can be used to derive polynomial-length scPCPPs for such statements (see Section 6). Therefore, we can replace the nearly-linear length PCPPs that are used in [1] with our polynomial length scPCPPs, while preserving the functionality of relaxed local decoding and keeping the total length of the construction nearly-linear. (See Section 4 for a more detailed high-level description of our approach, followed by a full proof in Section 4.2.)

Recapping, so far our construction is as follows. Let  $C$  be a good linear code and  $d \in \mathbb{N}$  be a sufficiently large constant. Each codeword of our code consists of the following equal-length parts: (1) repetitions of the tensor codeword  $C^{\otimes d}(x)$  that encodes the message  $x$ ; (2) repetitions of the explicitly given message  $x$ ; and (3) scPCPPs for small statements (specifically, regarding the consistency of each point in the tensor  $C^{\otimes d}(x)$  with each line that passes through it), which are used to ascertain the consistency of each individual bit in the message  $x$  with the codeword  $C^{\otimes d}(x)$ .<sup>8</sup>

Finally, we augment the aforementioned construction with a forth and last part that allows us to obtain *strong* local testability. The naive approach is to append a scPCPP

<sup>7</sup> Axis-parallel lines in high-dimensional tensors simply generalize the notion of rows and columns in  $n \times n$  matrices.

<sup>8</sup> We remark that the actual construction differs slightly from the above in that, for convenience, we use *systematic* tensor codes that contain the message explicitly in the encoding, instead of providing repetitions of the message as a part of the code.

that ascertains the validity of all three parts of our code. However, since the length of our scPCPPs is polynomial in the length of the statement, this approach would yield codes with long (polynomial) length. Instead, recall that we can (strongly) test the consistency of the first two parts via the third part (which is also strongly testable, since it is a scPCPP). Thus, in order to obtain strong local testability it suffices to ascertain that the first part is a valid codeword of  $C^{\otimes d}$  using scPCPPs. Luckily, tensor codes also satisfy the *robustness* feature, which allows us to ascertain the validity of an entire codeword of  $C^{\otimes d}$  by ascertaining the validity of small parts of the codeword. Detail follows.

Loosely speaking, a code is said to be *robust* if the corruption in a random “local view” of a codeword is proportional to the corruption in the entire codeword. In more detail, we use a recent result of Viderman [22] (building on [2]) that states that the corruption in a random 2-dimensional (axis-parallel) plane of a corrupted codeword of a binary tensor code  $C^{\otimes d}$  (where  $d \geq 3$ ) is proportional to the corruption in the entire codeword. This feature allows us to ascertain the validity of the first part (i.e., the tensor codeword  $C^{\otimes d}(x)$ ) by only providing scPCPPs for short statements that refer to 2-dimensional planes in  $C^{\otimes d}(x)$ . (See Section 5 for a more detailed high-level description, followed by a full proof.)

## 1.5 Applications to Property Testing

As an application of our main result (Theorem 1.1) we improve on the best known separation result (due to [15]) between the complexity of *decision* and *verification* in the setting of *property testing*.

The study of property testing, initiated by Rubinfeld and Sudan [20] and Goldreich, Goldwasser and Ron [11], considers highly-efficient randomized algorithms that solve approximate decision problems, while only inspecting a small fraction of the input. Recently, Gur and Rothblum [15] initiated the study of *MA proofs of proximity* (hereafter *MAPs*), which can be viewed as the NP analogue of property testing. They reduced the task of separating the power of property testers and *MAPs* to the design of very local codes, both in terms of testability and decodability. Furthermore, they noticed that for such a separation, *relaxed decodability* would suffice.

Gur and Rothblum used several weaker codes to obtain weaker separation results than the one we obtain here. Specifically, they either show a smaller gap between the query complexity of testers and *MAPs*, or show a separation for a limited range of the proximity parameter. In contrast, by plugging-in the code of Theorem 1.1, we obtain the best known (exponential) separation result between the power of *MAPs* and property testers.

► **Theorem 1.3 (Informal).** *There exists a property that requires  $n^{0.999}$  queries for every property tester but has an *MAP* that uses a proof of logarithmic length and makes  $\text{poly}(1/\varepsilon)$  queries.*

For more information regarding this application, we refer the reader to Section 7.

## 1.6 Organization

In Section 2 we provide the preliminaries. In Section 3 we describe the construction of the codes that establish Theorem 1.1. In Section 4 and Section 5 we establish the relaxed local decodability and strong local testability (respectively) of the codes. In Section 6 we construct the scPCPPs needed for our construction, and finally, in Section 7 we present an application of our codes for property testing.

## 2 Preliminaries

We start with some general notation. We denote by  $[n]$  the set of numbers  $\{1, 2, \dots, n\}$ . For  $i \in [n]$  and for  $x \in \{0, 1\}^n$ , denote by  $x_i$  the  $i^{\text{th}}$  bit of  $x$ . For  $x, y \in \{0, 1\}^n$ , we denote by  $\Delta(x, y)$  the Hamming distance between  $x$  and  $y$ , and denote by  $\delta(x, y)$  the *relative* (Hamming) distance between  $x$  and  $y$ , i.e.,  $\delta(x, y) = \Delta(x, y)/n$ . We say that  $x$  is  $\delta$ -close to (respectively,  $\delta$ -far from)  $y$  if the relative distance between  $x$  and  $y$  is at most  $\delta$  (respectively, at least  $\delta$ ).

Given a set  $S$ , we denote by  $s \in_R S$  the distribution that is obtained by selecting uniformly at random  $s \in S$ . For a randomized algorithm  $A$ , we write  $\Pr_A[\cdot]$  (or  $\mathbb{E}_A[\cdot]$ ) to state that the probability (or expectation) is over the internal randomness of the algorithm  $A$ .

### (Non) Uniformity

Throughout this paper, for the simplification of the presentation, we formally treat algorithms (testers, decoders, and verifiers) as (non-uniform) *polynomial-size circuits*. We note, however, that all of our algorithms can be made uniform by making straightforward modifications. Furthermore, it will be convenient for us to view the length  $n \in \mathbb{N}$  of objects as fixed. We note that although we fix  $n$ , it should be viewed as a generic parameter, and so we allow ourselves to write asymptotic expressions such as  $\text{poly}(n)$ ,  $O(n)$ , etc. In contrast, when we say that something is a constant, we mean that it is independent of the length parameter  $n$ .

### 2.1 Error Correcting Codes

Let  $k, n \in \mathbb{N}$ . A binary linear code  $C : \{0, 1\}^k \rightarrow \{0, 1\}^n$  of distance  $d$  is a linear mapping over  $\text{GF}(2)$ , which maps messages to codewords, such that the Hamming distance between any two codewords is at least  $d = d(n)$ . The relative distance of  $C$ , denoted by  $\delta(C)$ , is given by  $d/n$ . The length of a code is  $n = n(k)$ . By slightly abusing notation, we say that we can construct a code  $C$  with nearly linear length if for any constant  $\alpha > 0$  we can construct a code  $C : \{0, 1\}^k \rightarrow \{0, 1\}^n$ , where  $n = k^{1+\alpha}$ . For any  $x \in \{0, 1\}^n$ , denote the relative distance of  $x$  to the code  $C$  by  $\delta_C(x) = \min_{y \in C} \{\delta(x, y)\}$ .

We say that  $C$  is *systematic*, if the first  $k$  bits of every codeword of  $C$  contain the message; that is, if for every  $x \in \{0, 1\}^k$  and every  $i \in [k]$  it holds that  $C(x)_i = x_i$ . Since  $C$  is a linear code, we may assume without loss of generality that it is systematic.

### 2.2 Local Testability and Decodability

Following the discussion in the introduction, *strong* locally testable codes are defined as follows.

► **Definition 2.1** (strong-LTC). A code  $C : \{0, 1\}^k \rightarrow \{0, 1\}^n$  is a **strong-LTC**, if there exists a probabilistic algorithm (tester)  $T$  that, given oracle access to  $w \in \{0, 1\}^n$ , makes  $O(1)$  queries to  $w$ , and satisfies:

1. **Completeness:** For any codeword  $w$  of  $C$  it holds that  $T^w = 1$ .
2. **Strong Soundness:** For all  $w \in \{0, 1\}^n$ ,

$$\Pr_T[T^w = 0] \geq \text{poly}(\delta_C(w)).$$

We say that a tester makes *nearly-uniform* queries if it queries each bit in the (alleged) codeword input  $w \in \{0, 1\}^n$  with probability  $\Theta(1/n)$ .

Following the discussion in the introduction, *relaxed* locally decodable codes are defined as follows.

► **Definition 2.2** (relaxed-LDC). A code  $C : \{0, 1\}^k \rightarrow \{0, 1\}^n$  is a relaxed-LDC if there exists a constant  $\delta_{\text{radius}} \in (0, \delta(C)/2)$ , a constant  $\rho > 0$  and a probabilistic algorithm (decoder)  $D$  that, given oracle access to  $w \in \{0, 1\}^n$  and explicit input  $i \in [k]$ , makes  $O(1)$  queries to  $w$ , and satisfies:

1. **Completeness:** For any  $i \in [k]$  and  $x \in \{0, 1\}^k$  it holds that  $D^{C(x)}(i) = x_i$ .
2. **Relaxed Soundness:** For any  $i \in [k]$  and any  $w \in \{0, 1\}^n$  that is  $\delta_{\text{radius}}$ -close to a codeword  $C(x)$ ,<sup>9</sup> it holds that

$$\Pr_D[D^w(i) \in \{x_i, \perp\}] \geq 2/3.$$

3. **Success Rate:** For every  $w \in \{0, 1\}^n$  that is  $\delta_{\text{radius}}$ -close to a codeword  $C(x)$ , and for at least a  $\rho$  fraction of the indices  $i \in [k]$ , with probability at least  $2/3$  the decoder  $D$  outputs the  $i^{\text{th}}$  bit of  $x$ . That is, there exists a set  $I_w \subseteq [k]$  of size at least  $\rho k$  such that for every  $i \in I_w$  it holds that  $\Pr_D[D^w(i) = x_i] \geq 2/3$ .

We remark that our definition is slightly *stronger* than the one given in [1] as we require *perfect* completeness (i.e., that the decoder *always* outputs the correct value given oracle access to a valid codeword of the code  $C$ ).

### 2.3 Tensor Codes

Tensor codes are defined as follows.

► **Definition 2.3.** Let  $C : \{0, 1\}^k \rightarrow \{0, 1\}^n$  be a linear code. The tensor code  $C \otimes C : \{0, 1\}^{k^2} \rightarrow \{0, 1\}^{n^2}$  is the code whose codewords consists of all  $n \times n$  matrices such that each axis-parallel line (i.e., a row or a column) in the matrix is a codeword of  $C$ . Similarly, given  $d \in \mathbb{N}$ , the tensor code  $C^{\otimes d} : \{0, 1\}^{k^d} \rightarrow \{0, 1\}^{n^d}$  is the code whose codewords consists of all  $d$ -dimensional tensors such that each axis-parallel line in the tensor is a codeword of  $C$ .

It is well-known that for every  $d \in \mathbb{N}$  the tensor code  $C^{\otimes d}$  is a linear code with relative distance  $\delta(C)^d$  (see e.g., [2]). Given a message  $x \in \{0, 1\}^{k^d}$  and coordinate  $\bar{i} = (\bar{i}_1, \dots, \bar{i}_d) \in [n]^d$ , we denote the value of  $C^{\otimes d}(x)$  at coordinate  $\bar{i}$  by  $C^{\otimes d}(x)_{\bar{i}}$ .

► **Remark.** By the definition of tensor codes, if a linear code  $C$  is systematic, then the tensor code  $C^{\otimes d}$  is also a systematic code;<sup>10</sup> that is, for every  $x \in \{0, 1\}^{k^d}$  and  $\bar{i} \in [k]^d$  it holds that  $C^{\otimes d}(x)_{\bar{i}} = x_{\bar{i}}$ .

Next, we provide notations for the restriction of tensors to lines and planes. We start by defining axis-parallel lines.

► **Definition 2.4** (Axis-Parallel Lines). For  $j \in [d]$  and  $\bar{i} = (i_1, \dots, i_d) \in [n]^d$ , we denote by  $\ell_{j, \bar{i}}$  the  $j^{\text{th}}$  axis-parallel line passing through  $\bar{i}$ . That is,

$$\ell_{j, \bar{i}} = \{(i_1, \dots, i_{j-1}, x, i_{j+1}, \dots, i_d)\}_{x \in [n]}.$$

We denote by  $\text{Lines}(n, d)$  the *multi-set* that contains all axis-parallel lines that pass through each point  $\bar{i} \in [n]^d$ .<sup>11</sup> That is,  $\text{Lines}(n, d) = \{\ell_{j, \bar{i}}\}_{\bar{i} \in [n]^d, j \in [d]}$ . Lastly, given a tensor  $w \in \{0, 1\}^{n^d}$  we denote by  $w|_{\ell_{i, j}} \in \{0, 1\}^n$  the restriction of  $w$  to the line  $\ell_{i, j}$ , i.e., the  $j^{\text{th}}$  axis-parallel line that passes through  $\bar{i}$ .

<sup>9</sup> Note that since  $\delta_{\text{radius}} < \delta(C)/2$ , for every  $x \in \{0, 1\}^n$  that is  $\delta_{\text{radius}}$ -close to  $C$  there exists a *unique* codeword  $x'$  of  $C$  such that  $x$  is  $\delta_{C'}(x)$ -close to  $x'$ .

<sup>10</sup> We view the restriction of the tensor  $C^{\otimes d}$  to the coordinates in  $[k]^d$  as the prefix of  $C^{\otimes d}$ .

<sup>11</sup> Note that each axis-parallel line in  $\{0, 1\}^{n^d}$  appears  $n$  times in  $\text{Lines}(n, d)$ .

Next, we define axis-parallel planes.

► **Definition 2.5** (Axis-Parallel (2-dimensional) Planes). For  $j_1 < j_2 \in [d]$  and  $\bar{i} = (i_1, \dots, i_d) \in [n]^d$ , we denote by  $p_{j_1, j_2, \bar{i}}$  the  $(j_1, j_2)^{\text{th}}$  axis-parallel plane passing through the point  $\bar{i}$ . That is

$$p_{j_1, j_2, \bar{i}} = \{(i_1, \dots, i_{j_1-1}, x_1, i_{j_1+1}, \dots, i_{j_2-1}, x_2, i_{j_2+1}, \dots, i_d)\}_{x_1, x_2 \in [n]}.$$

We denote by  $\text{Planes}(n, d)$  the set of all (distinct) axis-parallel planes in all directions in  $\{0, 1\}^{n^d}$ .<sup>12</sup> Lastly, for a tensor  $w \in \{0, 1\}^{n^d}$  and a plane  $p \in \text{Planes}(n, d)$  we denote by  $w|_p \in \{0, 1\}^{n^2}$  the restriction of  $w$  to the coordinates in the plane  $p$ .

Throughout this work we deal with axis-parallel lines (respectively, axis-parallel planes); hence, for brevity, we will sometimes refer to an *axis-parallel line* (respectively, *axis-parallel plane*) simply as a *line* (respectively, *plane*). We remark that the multi-set  $\text{Lines}(n, d)$  contains  $d \cdot n^d$  lines and the set  $\text{Planes}(n, d)$  contains  $\binom{d}{2} \cdot n^{d-2}$  planes. We omit the parameters  $n$  and  $d$  when they are clear from the context.

## Testing Tensor Codes

The next theorem, which is implicit in [22], shows that for every  $d \geq 3$  and every linear code  $C$ , testing the tensor-code  $C^{\otimes d}$  can be reduced to testing whether a random plane in  $C$  is a codeword of  $C^{\otimes 2}$ .

► **Theorem 2.6.** *Let  $C$  be a linear binary code and  $d \geq 3$  an integer. Then, there exists a constant  $c_{\text{robust}} \in (0, 1)$  such that for every tensor  $w \in \{0, 1\}^{n^d}$  it holds that*

$$\mathbb{E}_{p \in_R \text{Planes}} [\delta(w|_p, C^{\otimes 2})] > c_{\text{robust}} \cdot \delta_{C^{\otimes d}}(w).$$

Specifically, in [22, Theorem A.5] it is shown that for  $d \geq 3$ , if a codeword  $w$  of a tensor code  $C^{\otimes d}$  is corrupted, then the corruption in a random  $(d-1)$ -dimensional subplane of  $w$  is proportional to the corruption in the entire tensor  $w$ . By applying this result recursively (a constant number of times), we obtain Theorem 2.6. For completeness, we provide the proof of Theorem 2.6 in Appendix C.

## 2.4 PCPs of Proximity

Strong canonical PCPs of proximity were defined as follows in [13, Section 5.3].

► **Definition 2.7** (scPCPPs). Let  $V$  be a probabilistic algorithm (verifier) that is given *oracle* access to an input  $x \in \{0, 1\}^n$  and *oracle* access to a proof  $\pi \in \{0, 1\}^{\ell(n)}$ , where  $\ell : \mathbb{N} \rightarrow \mathbb{N}$  satisfies  $\ell(n) \leq \exp(\text{poly}(n))$ . We say that  $V$  is a **strong (canonical) PCPP verifier** for language  $L$  if it makes  $O(1)$  queries and satisfies the following two conditions:

- **Canonical Completeness:** For all  $x \in L$ , there exists a *unique canonical proof* for  $x$ , denoted  $\pi_{\text{canonical}}(x)$ , such that the verifier always accepts the pair  $(x, \pi_{\text{canonical}}(x))$ ; i.e.,  $V^{x, \pi_{\text{canonical}}(x)} = 1$ .

<sup>12</sup> Unlike the *multi-set*  $\text{Lines}(n, d)$ , which contains  $n$  copies of each line, there is no redundancy in the *set*  $\text{Planes}(n, d)$ .



- **Strong Canonical Soundness:** For any input  $x' \in \{0, 1\}^n$  and proof  $\pi' \in \{0, 1\}^{\ell(|x|)}$  the verifier rejects with probability at least  $\text{poly}(\delta_{\text{PCPP}}(x', \pi'))$ , where

$$\delta_{\text{PCPP}}(x', \pi') \triangleq \min_{x \in \{0, 1\}^n} \left\{ \max \left( \frac{\Delta(x, x')}{n}; \frac{\Delta(\pi_{\text{canonical}}(x), \pi')}{\ell(n)} \right) \right\}, \quad (1)$$

where for any  $x \notin L$  we define  $\pi_{\text{canonical}}(x) = \lambda$  and say that any  $\pi'$  is 1-far from  $\lambda$ . We say that a scPCPP verifier makes *nearly-uniform* queries if it queries each bit in the input  $x$  with probability  $\Theta(1/|x|)$  and queries each bit in the proof  $\pi(x)$  with probability  $\Theta(1/|\pi|)$ .

We stress that these scPCPPs have *one-sided error* (i.e., they always accept inputs in  $L$  coupled with their canonical proofs). Note that the *canonical* aspect is reflected in the dependence of  $\delta_{\text{PCPP}}(x', \pi')$  on  $\Delta(\pi_{\text{canonical}}(x), \pi')$ , whereas the *strong-soundness* aspect is reflected in the tight relation between the rejection probability and  $\delta_{\text{PCPP}}(x', \pi')$ .

### 3 The Main Construction

In this section we describe our construction of a family of binary linear codes that are both (constant-query) relaxed-LDCs and strong-LTCs with constant relative distance and nearly-linear length. Our codes rely heavily on special-purpose *strong canonical* PCPs of *proximity* (with polynomial proof length), which we construct in Section 6, and so, we start by stating these scPCPPs. Our first family of scPCPPs is for good linear codes.

► **Theorem 3.1** (scPCPPs for good codes). *Let  $C : \{0, 1\}^k \rightarrow \{0, 1\}^n$  be a linear code with constant relative distance and linear length. Then, there exists a scPCPP for codewords of  $C$  (i.e., for the set  $\{C(x)\}_{x \in \{0, 1\}^k}$ ). Furthermore, the proof length of the scPCPP is  $\text{poly}(n)$ , the scPCPP verifier makes nearly-uniform queries, and the canonical scPCPP proofs are linear (over  $\text{GF}(2)$ ).*

As a corollary of Theorem 3.1, we obtain a family of scPCPPs for *half-spaces* of any good linear code. That is, scPCPPs that ascertain membership in the set of all codewords wherein one given location is set to a specific value (for example, all codewords that have 1 in their first location).

► **Theorem 3.2** (scPCPPs for half-spaces of good codes). *Let  $C : \{0, 1\}^k \rightarrow \{0, 1\}^n$  be a linear code with constant relative distance and linear length. Let  $i \in [k]$  be a location in a message and  $b \in \{0, 1\}$  a bit. Then, there exists a scPCPP for  $C_{i,b}$ , where  $C_{i,b}$  is the set of all codewords  $w$  of  $C$  such that the  $i^{\text{th}}$ -bit of  $w$  equals  $b$  (i.e.,  $w_i = b$ ). Furthermore, the proof length of the scPCPP is  $\text{poly}(n)$ , the scPCPP verifier makes nearly-uniform queries, and the scPCPP proofs are linear (over  $\text{GF}(2)$ ).*

See Section 6 for the full proofs of Theorems 3.1 and 3.2. Equipped with the foregoing scPCPPs, we describe the construction of our code, which consists of three parts. (See Section 2 for relevant notation.)

#### Tensor code part

Let  $C_0 : \{0, 1\}^k \rightarrow \{0, 1\}^n$  be a systematic linear code with linear length (i.e.,  $n = \Theta(k)$ ) and constant relative distance  $0 < \delta(C_0) < 1$ . Let  $d \geq 3$  be a sufficiently large constant (to be determined later). Let  $C \triangleq (C_0)^{\otimes d} : \{0, 1\}^{k^d} \rightarrow \{0, 1\}^{n^d}$  be the  $d$ -tensor product of  $C_0$ . By Section 2.3, since  $C_0$  is systematic, then  $C$  is also systematic. Recall that  $\delta(C) = \delta(C_0)^d$ , hence  $\delta(C)$  is a constant.

We augment the code  $C$  with scPCPPs that ascertain the validity of each *plane* in  $C$  (using Theorem 3.1) and scPCPPs that ascertain the consistency of each bit in  $C$  with each line that passes through it (using Theorem 3.2). Details follow.

### Plane scPCPPs part

Let  $C(x)$  be a codeword of the tensor code  $C$ . For every plane  $\mathfrak{p}$  in the tensor  $C(x)$  we use our scPCPPs *for good codes* to prove that the restriction of  $C(x)$  to the plane  $\mathfrak{p}$  (denoted by  $C(x)|_{\mathfrak{p}}$ ) is a codeword of  $C_0^{\otimes 2}$ . Specifically, for a codeword  $w$  of  $C_0^{\otimes 2}$  we denote by  $\pi_{\text{plane}}(w)$  the corresponding canonical proof for the scPCPP verifier of Theorem 3.1. Then, for every message  $x \in \{0, 1\}^{k^d}$  we define  $\pi_{\text{planes}}(x)$  as the sequence of the canonical proofs for *all* planes in  $C(x)$ ; that is,

$$\pi_{\text{planes}}(x) = \{\pi_{\text{plane}}(C(x)|_{\mathfrak{p}})\}_{\mathfrak{p} \in \text{Planes}},$$

where  $\text{Planes}$  is the set of *all* (2-dimensional) axis-parallel planes in  $\{0, 1\}^{n^d}$  (see Definition 2.5).

We append  $\pi_{\text{planes}}(x)$  to the codeword  $C(x)$ . Note that

$$|\pi_{\text{planes}}(x)| = \binom{d}{2} n^{d-2} \cdot |\pi_{\text{plane}}(C(x)|_{\mathfrak{p}})| \leq n^{d+O(1)}.$$

We stress that the constant in the  $O(1)$  notation does *not* depend on  $d$ . These scPCPPs will be used for the local testability of our code (see Section 5).

### Point-line scPCPPs part

Let  $C(x)$  be a codeword of the tensor code  $C$ . For every point  $\bar{i} = (i_1, \dots, i_d) \in [n]^d$  and every direction  $j \in [d]$  we use our scPCPPs *for half-spaces of good codes* to prove that the restriction of  $C(x)$  to the line that passes through point  $\bar{i}$  in direction  $j$  (denoted by  $C(x)|_{\ell_{j,\bar{i}}}$ ) is a codeword of  $C_0$  that is consistent with value of  $C(x)$  at point  $\bar{i}$ .<sup>13</sup> Specifically, for a codeword  $w$  of  $C_0$  and index  $s \in [n]$  we denote by  $\pi_{\text{line}}(w, s)$  the canonical proof for the scPCPP verifier of Theorem 3.2 (which corresponds to codewords of  $C_0$  whose  $s^{\text{th}}$ -bit equals to  $w_s$ ). Then, for every message  $x \in \{0, 1\}^{k^d}$  we define  $\pi_{\text{lines}}(x)$  as the set of the canonical proofs for *all* lines passing through each point in  $C(x)$ ; that is,

$$\pi_{\text{lines}}(x) = \{\pi_{\text{line}}(C(x)|_{\ell_{j,\bar{i}}}, i_j)\}_{\ell_{j,\bar{i}} \in \text{Lines}},$$

where  $\text{Lines} = \{\ell_{j,\bar{i}}\}_{\bar{i} \in [n]^d, j \in [d]}$ , as in Definition 2.4 (i.e., the set  $\text{Lines}$  contains all axis-parallel lines that pass through each point  $\bar{i} \in [n]^d$ ).

We append  $\pi_{\text{lines}}(x)$  to the codeword  $C(x)$ . Note that  $|\pi_{\text{lines}}(x)| = d \cdot n^d \cdot |\pi_{\text{line}}(C(x)|_{\ell})| \leq n^{d+O(1)}$ , where the constant in the  $O(1)$  notation does *not* depend on  $d$ . These scPCPPs will be used for the relaxed local decodability of our code (see Section 4).

### Putting it all together

Our construction is obtained by combining the tensor codeword  $C(x)$  with  $\pi_{\text{lines}}(x)$  and  $\pi_{\text{planes}}(x)$ , while ensuring that the three parts are of equal length. That is, for  $k' = k^d$  define

<sup>13</sup>Note that the  $\bar{i}^{\text{th}}$ -bit of  $C(x)$  is, in fact, the  $i_j^{\text{th}}$ -bit of the line  $C(x)|_{\ell_{j,\bar{i}}}$ .



$C' : \{0, 1\}^{k'} \rightarrow \{0, 1\}^{n'}$  as follows.

$$C'(x) \triangleq \left( (C(x))^{t_1}, (\pi_{\text{lines}}(x))^{t_2}, (\pi_{\text{planes}}(x))^{t_3} \right)$$

where  $t_1, t_2$  and  $t_3$  are the *minimal* integers such that  $|C(w)|^{t_1} = |\pi_{\text{lines}}(w)|^{t_2} = |\pi_{\text{planes}}(w)|^{t_3}$ .<sup>14</sup>

### Length and relative-distance of $C'$

For sufficiently large  $d$  the length of  $C'$  is nearly-linear. To this end, observe that for every  $x \in \{0, 1\}^{k^d}$  it holds that  $|C(x)| = n^d$ ,  $|\pi_{\text{lines}}(x)| \leq \text{poly}(n)$  and  $|\pi_{\text{planes}}(x)| \leq \text{poly}(n^2)$ . Hence, for every constant  $\alpha > 0$ , there exists some constant  $d > 0$  so that

$$n' = n^{d+O(1)} = (O(1) \cdot k)^{d+O(1)} \leq (k')^{1+\alpha}.$$

The code  $C'$  has constant relative distance since the relative distance of  $C$  (denoted by  $\delta(C)$ ) is constant, and since repetitions of  $C$  constitute a third of the length of  $C'$ ; that is,  $\delta(C') \geq \frac{\delta(C)}{3}$ . In the next sections we prove the following theorem.

► **Theorem 1.1** (restated). *For every constant  $\alpha > 0$ , there exists some constant  $d \geq 0$  so that the code  $C' : \{0, 1\}^{k'} \rightarrow \{0, 1\}^{n'}$ , as defined above, is a linear binary code that is a relaxed-LDC and a strong-LTC with constant relative distance.*

Specifically, in Section 4 we prove the relaxed-LDC feature of  $C'$ , and in Section 5 we prove the strong-LTC feature of  $C'$ .

### (Alleged) Codeword Notations

Consider an arbitrary string  $w \in \{0, 1\}^{n'}$  (which we think of as an alleged codeword). We view  $w$  as a string composed of three parts (analogous to the three parts of the construction above):

1.  $\bar{c} = (c_1, \dots, c_{t_1})$ : the  $t_1$  alleged repetitions of the tensor code part.
2.  $\bar{p}^{\text{lines}} = (\bar{p}_1^{\text{lines}}, \dots, \bar{p}_{t_2}^{\text{lines}})$ : the  $t_2$  alleged repetitions of the scPCPP proofs for all the point-line pairs (i.e., lines passing through all coordinates in all directions). For every  $i \in [t_2]$ , the string  $\bar{p}_i^{\text{lines}}$  consists of scPCPP proofs for every point-line pair, i.e.,  $\bar{p}_i^{\text{lines}} = \{p_i^\ell\}_{\ell \in \text{Lines}}$ .
3.  $\bar{p}^{\text{planes}} = (\bar{p}_1^{\text{planes}}, \dots, \bar{p}_{t_3}^{\text{planes}})$ : the  $t_3$  alleged repetitions of the scPCPP proofs for all the (2-dimensional) planes. For every  $i \in [t_3]$ , the string  $\bar{p}_i^{\text{planes}}$  consists of scPCPP proofs for every plane, i.e.,  $\bar{p}_i^{\text{planes}} = \{p_i^p\}_{p \in \text{Planes}}$ .

## 4 Establishing the Relaxed-LDC Property

In this section we prove that the code  $C'$ , which was defined in Section 3, is a relaxed locally decodable code.

► **Theorem 4.1.** *The code  $C' : \{0, 1\}^{k'} \rightarrow \{0, 1\}^{n'}$  is a relaxed-LDC.*

<sup>14</sup>Ignoring integrality issues, we can say that we “blow” the lengths of the two shorter parts to match the length of the longest part, which (in case of our implementation of the scPCPPs) is the part of the *plane* scPCPPs. Hence, actually,  $t_3 = 1$ .

In order to prove Theorem 4.1, it would be convenient to use an alternative definition of relaxed-LDCs, which implies the standard definition (Definition 2.2) by applying known transformations. Specifically, in Section D (following [1, Section 4.2]) we show that it suffices to relax the soundness parameter in Definition 2.2 to  $\Omega(1)$  (instead of  $2/3$ ), and replace the *success rate* condition with the following *average smoothness* condition. Loosely speaking, *average smoothness* requires that the decoder makes nearly uniform queries on average (over all indices to be decoded). By the foregoing, to prove Theorem 4.1 it suffices to show that the code  $C'$  satisfies the following definition.

► **Definition 4.2** (*Modified relaxed-LDCs*). A code  $C : \{0, 1\}^k \rightarrow \{0, 1\}^n$  is a *modified relaxed-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  $w \in \{0, 1\}^n$  and explicit input  $i \in [k]$ , makes  $q = O(1)$  queries to  $w$ , and satisfies:

1. **Completeness:** For any  $i \in [k]$  and  $x \in \{0, 1\}^k$  it holds that  $D^{C(x)}(i) = x_i$ .
2. **Modified Relaxed Soundness:** For any  $i \in [k]$  and any  $w \in \{0, 1\}^n$  that is  $\delta_{\text{radius}}$ -close to a codeword  $C(x)$  it holds that

$$\Pr_D [D^w(i) \in \{x_i, \perp\}] = \Omega(1).$$

where  $\delta_{\text{radius}} \in (0, \delta(C')/2)$ , the decoding radius of  $C$ , is a universal constant, to be determined later.

3. **Average Smoothness:** for every  $w \in \{0, 1\}^n$  and  $v \in [n]$ ,

$$\Pr_{i,j,r} [\mathcal{D}^w(i, j, r) = v] < \frac{2}{n},$$

where  $\mathcal{D}^w(i, j, r)$  denotes the distribution of the  $j^{\text{th}}$  query of the decoder  $D^w$  on coordinate  $i$  and coin tosses  $r$ , where the probability is taken uniformly over all possible choices of  $i \in [k]$ ,  $j \in [q]$ , and coin tosses  $r$ .

We remark that in [1, Section 4.2], the definition of average smoothness also requires a matching lower bound, i.e., the decoder should satisfy  $\frac{1}{2n} < \Pr_{i,j,r} [\mathcal{D}^w(i, j, r) = v] < \frac{2}{n}$ . However, for our applications it suffices to only require the upper bound. We note that the lower bound can be easily obtained by adding (random) dummy queries.

We start by showing a decoder that satisfies the first two aforementioned conditions (i.e., the *completeness* condition and the *modified relaxed soundness*). Next, in Section 4.3 we show how to obtain a related decoder that also satisfies the *average smoothness* condition.

## The Setting

Consider an arbitrary input  $w \in \{0, 1\}^{n'}$  such that  $0 \leq \delta_{C'}(w) < \delta_{\text{radius}}$ . We view  $w$  as a string composed of three parts as in Section 3, i.e.,  $w = (\bar{c}, \bar{p}^{\text{lines}}, \bar{p}^{\text{planes}})$ . We stress that any part of  $w$  might suffer from corruptions, and so, we have to be able to decode correctly assuming that not too many corruptions have occurred (i.e., less than  $\delta_{\text{radius}}$  fraction). Denote by  $x$  the unique string such that  $w$  is  $\delta_{C'}(w)$ -close to  $C(x)$  (see footnote 9).

## High-Level Idea

Recall that a valid codeword of  $C'$  consists of three (repeated) parts: (1) a systematic tensor code  $C$ , (2) point-line scPCPPs, and (3) plane scPCPPs. Our general approach is to decode according to the prefix of the first part (which allegedly contains the message  $x$  explicitly (since we use a systematic code), and to use the second part to ensure that each bit in

message  $x$  is consistent with the rest of the (tensor) codeword  $C(x)$ . (The third part is not used here; it is only used for the testability of the code.) Thus, the task of (relaxed) decoding the  $i^{\text{th}}$  bit of the message is reduced to verifying that the explicitly given value of the  $i^{\text{th}}$  bit of the message is consistent with the rest of the codeword.

Towards this end, recall that the second part of each codeword contains scPCPPs that ascertain the consistency of each bit in the tensor with each line that passes through it, but not consistency with the entire tensor. Therefore, in order to verify the consistency of each message bit with the entire codeword, our decoder uses a feature of tensor codes, which we call *local propagation*. This feature allows us to verify the consistency of a single message bit with the entire codeword by verifying the consistency of a carefully chosen sequence of  $d$  point-line pairs (using the *point-line* scPCPP). Details follow.

Loosely speaking, the *local propagation* feature of tensor codes implies that if one corrupts a single point in a codeword and attempts to keep most local views (say, lines in the tensor) consistent with this corruption, then a chain of highly structured modifications must be made that causes the “corruption” to propagate throughout the entire tensor. This is best exemplified by our decoder, which is tailored to take advantage of the foregoing phenomena.

Our decoder is given a coordinate  $\bar{i} = (i_1, \dots, i_d) \in [k]^d$  and oracle access to an alleged codeword  $w$  as above. The decoder looks for “inconsistencies” in  $w$  and if it finds any, it outputs  $\perp$ . Otherwise, it simply outputs  $w_{\bar{i}}$  (which should contain the  $\bar{i}^{\text{th}}$  bit of the message). Since our base code  $C_0$  has constant relative distance, in order to “corrupt” the point  $\bar{i}$  in the tensor code without causing the lines that pass through  $\bar{i}$  to be inconsistent with the corrupted value at  $\bar{i}$ , one has to corrupt a *constant fraction* of each line on which  $\bar{i}$  resides. Thus, our decoder uses the scPCPPs to verify that a line  $\ell$  that passes through  $\bar{i}$  is consistent with the value at  $\bar{i}$ , assuring that a constant fraction of many lines on which  $\bar{i}$  resides is corrupted.

Similarly, in order to “corrupt” a constant fraction of the line  $\ell$  in the tensor codeword without causing inconsistency between the corrupted points in  $\ell$  and the lines that pass through these corrupted points, one has to change a *constant fraction* of each line that passes through a corrupted point in  $\ell$  (therefore, corrupting a *constant fraction* of each *plane* wherein the line  $\ell$  resides). Thus, our decoder uses the scPCPPs to verify that the line that passes through a random point  $\bar{i}'$  in  $\ell$  (which is corrupted with probability  $\Omega(1)$ ) is consistent with the value at  $\bar{i}'$ , assuring that a constant fraction of many planes on which line  $\ell$  resides were corrupted.

Thus, if the  $\bar{i}^{\text{th}}$  point of the tensor codeword (i.e., the bit we wish to decode) is *corrupted*, then by iteratively continuing this procedure  $d$  times, and only performing  $d$  point-line consistency tests, the decoder can detect the corruption in  $\bar{i}$  with high probability, unless a large fraction of the codeword is corrupted (i.e., the corruption at a single point,  $\bar{i}$ , propagated to the entire tensor).

We remark that in the proof that  $C'$  is a relaxed-LDC we do not use the *strongness* and *canonicity* properties of the scPCPPs (they are only used to prove that  $C'$  is a strong-LTC). Furthermore, since in the following we only wish to present a decoder satisfies Condition 1 and 2 of Definition 4.2, we can allow the decoder to output a “don’t-know” symbol whenever the codeword is corrupted.<sup>15</sup> Thus, we are not concerned with corruptions in the scPCPP

<sup>15</sup>Recall that the *completeness* condition of Definition 4.2 requires the decoder to successfully decode valid codeword, and the *modified relaxed soundness* condition requires that the decoder does not make a mistake in the decoding with probability at least  $\Omega(1)$ . However, the decoder is allowed to output a “don’t-know” symbol with arbitrary probability on any (even on only slightly) corrupted codeword.

parts, since a corruption in these parts can only increase the rejection probability for strings that are not codewords. Regarding inputs that are legal codewords, there are no corruptions and hence, no “inconsistencies”. Thus, for legal codewords our tester will always output the correct value.

#### 4.1 Warm-up: Two-Dimensional Tensors

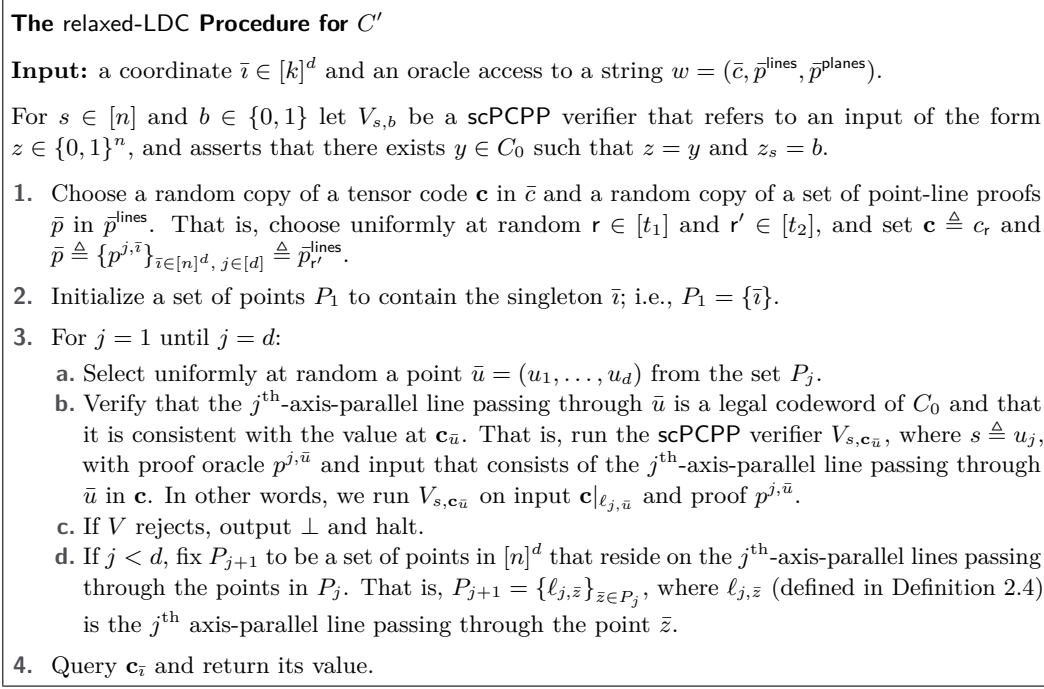
Before we proceed to prove Theorem 4.1, we sketch a proof for two-dimensional tensor codes; that is, when we set  $d = 2$  in the construction that appears in Section 3. In this warm-up, towards the end of simplifying the presentation, we make the following assumptions: We omit the third part of the codeword (i.e., the plane scPCPPs), and we omit the repetitions of the first and second parts of the code (i.e., the tensor code, and the point-line scPCPPs) and assume instead that the lengths of the first and the second parts are equal. We note that both assumptions can be easily removed (see Section 4.2 for details).

Let  $w = (c, p)$  be an alleged codeword that consists of two parts of equal length: (1)  $c$ , an alleged 2-dimensional tensor code  $C_0^{\otimes 2} : \{0, 1\}^{k^2} \rightarrow \{0, 1\}^{n^2}$ , and (2)  $p$ , a sequence of alleged scPCPPs for every pair of point  $\bar{i}$  in  $[n]^2$  and line  $\ell$  in  $C_0^{\otimes 2}$  that passes through  $\bar{i}$ ; each scPCPP ascertains that the line  $\ell$  is a codeword of  $C_0$  that is consistent with the value at the point  $\bar{i}$ .

Given a point  $\bar{i} = (i_1, i_2) \in [k]^2$ , the decoder first runs the point-line scPCPP that corresponds to  $\bar{i}$  and the line  $\ell_{1, \bar{i}} = \{(x, i_2)\}_{x \in [n]}$  passing through  $\bar{i}$  in direction “1” (i.e., parallel to the first axis), and outputs  $\perp$  if the scPCPP verifier rejected. Otherwise, the decoder picks a random point  $\bar{i}' = (i'_1, i'_2)$  on the line  $\ell_{1, \bar{i}}$ , runs the corresponding scPCPP for  $\bar{i}'$  and the line  $\ell_{2, \bar{i}'} = \{(i'_1, x)\}_{x \in [n]}$  that passes through  $\bar{i}'$  in direction “2”, and output  $\perp$  if the scPCPP verifier rejected. If none of the scPCPP verifiers rejected, the verifier outputs  $c_{\bar{i}}$ .

For the *completeness* condition, assume that the decoder is given a valid codeword. In this case, the first part is indeed a valid copy of  $C_0^{\otimes 2}(x)$ , and the second part consists of the canonical proofs for  $C_0^{\otimes 2}(x)$ . Hence, all of the scPCPP verifiers accept, and since  $C_0^{\otimes 2}(x)_{\bar{i}} = x_{\bar{i}}$ , the decoder succeeds in decoding  $x_{\bar{i}}$ .

For the (modified) *relaxed soundness* condition, assume that the decoder is given a corrupted codeword  $w = (c, p)$  that is  $\delta$ -close to a valid codeword  $C_0^{\otimes 2}(x)$ , where  $\delta \leq \delta_{\text{radius}}$  for a sufficiently small (constant) *decoding radius*  $\delta_{\text{radius}}$ . Note that if  $c_{\bar{i}} = x_{\bar{i}}$ , then the decoder satisfies the soundness condition (since it always outputs either  $x_i$  or  $\perp$ ); hence, we assume that  $c_{\bar{i}} \neq x_{\bar{i}}$ . In this case, when the decoder runs the scPCPP verifier for  $\bar{i}$  and (the restriction of  $c$  to)  $\ell_{1, \bar{i}}$  it does not reject (with high probability) only if  $C|_{\ell_{1, \bar{i}}}$  is “close” to a codeword of  $C_0$  that disagrees with  $c$  on  $\bar{i}$  (since the  $i_2^{\text{th}}$  bit of this codeword of  $C_0$  must be different than  $x_{\bar{i}}$ ). Since  $C_0$  is a code with constant relative distance, this implies that a constant fraction of the line  $\ell_{1, \bar{i}}$  must be corrupted (i.e., the restriction of  $c$  to the line  $\ell_{1, \bar{i}}$  is  $\Omega(1)$ -far from its corresponding line in  $C(x)$ ) for the scPCPP verifier to accept. Finally, if the decoder selected  $\bar{i}'$  that is one of the  $\Omega(n)$  corrupted points on  $\ell_{1, \bar{i}}$ , then by the same argument, a constant fraction points on the restriction of  $c$  to the line  $\ell_{2, \bar{i}'}$  (that passes through  $\bar{i}'$ ) must be corrupted. We deduce that in order to both scPCPP verifiers to accept (and hence defy the soundness condition),  $c$  must contain  $\Omega(n^2)$  corrupted points, i.e.,  $c$  should be  $\beta$ -far from  $C_0^{\otimes 2}(x)$  for some constant  $\beta$ . By fixing  $\delta_{\text{radius}} < \beta$ , we prevent this possibility.



■ **Figure 1** Relaxed local decoder  $D$  for  $C'$ .

## 4.2 The General Case

We proceed with the full proof that  $C'$  has a decoder that satisfies the first two conditions in the definition of a relaxed-LDC (i.e., the *completeness* and (*modified*) *relaxed soundness* conditions of Definition 4.2). We generalize the decoder of Section 4.1 to  $d$ -dimensional tensors and ensure it works without the assumptions that were made there for simplicity. The decoder  $D$  is formally described in Figure 1.

Let  $\bar{i} \in [k]^d$ . The *completeness* of the decoder is immediate from the construction: If the input is a codeword, i.e.,  $w = C'(x)$  and all of the scPCPPs proofs are the canonical proofs for  $C'(x)$  (i.e.,  $\bar{p}^{\text{lines}}$  and  $\bar{p}^{\text{planes}}$ ), then all of the executions of the scPCPP verifiers accept (since the scPCPP verifiers are with one-sided error). Recalling that, by definition,  $C(x)_{\bar{i}} = C_0^{\otimes d}(x)_{\bar{i}} = x_{\bar{i}}$ , the decoding procedure  $D^w(\bar{i})$  returns  $x_{\bar{i}}$  with probability 1, as required.

Next, we prove the (*modified*) *relaxed soundness* of the decoder. Let  $w \in \{0, 1\}^n$  be a corrupted codeword that is  $\delta_{\text{radius}}$ -close to a codeword  $C(x)$ , where  $\delta_{\text{radius}}$  is a sufficiently small constant, to be determined later. We partition the analysis into three cases (Claims 4.3 and 4.4 and Theorem 4.5) that we analyze in the rest of this section. We begin with the following two simple claims.

The first claim shows that probability  $\Omega(1)$ , the random copy  $\mathbf{c}$  in  $(c_1, \dots, c_{t_1})$  that is chosen in Step 1 cannot be “too far” from the codeword  $C(x)$ .

► **Claim 4.3.** *With probability at least  $1/4$ , the random copy  $\mathbf{c}$  is  $4\delta_{C'}(w)$ -close to  $C(x)$ , where  $\mathbf{c}$  is chosen uniformly at random from  $\bar{c}$ . That is,*

$$\Pr_{\mathbf{c} \in R(c_1, \dots, c_{t_1})} [\delta_C(\mathbf{c}) \leq 4\delta_{C'}(w)] \geq \frac{1}{4}.$$

**Proof.** Since  $|\bar{c}| = |\bar{p}^{\text{lines}}| = |\bar{p}^{\text{planes}}|$ , then  $\bar{c} = (c_1, \dots, c_{t_1})$  is  $3\delta_{C'}(w)$ -close to  $C(x)^{t_1}$ . This means that the expected relative distance of a random  $\mathbf{c} \in \{c_1, \dots, c_{t_1}\}$  from  $C(x)$  is at most  $3\delta_{C'}(w)$ . Hence, by Markov's inequality,  $\mathbf{c}$  is  $4\delta_{C'}(w)$ -far from  $C(x)$  with probability at most  $3/4$ .  $\blacktriangleleft$

Therefore, throughout the rest of the proof we fix a random copy  $\mathbf{c}$  and assume that it is  $4\delta_{C'}(w)$ -close to  $C(x)$ . This only costs us at most a constant factor in the success probability of the decoder. Having fixed  $\mathbf{c}$ , recall that for  $\bar{i} \in [n]^d$ , the notation  $\mathbf{c}_{\bar{i}}$  refers to the value of  $\mathbf{c}$  at point  $\bar{i}$ . The next claim shows that if the bit we are trying to decode is not “corrupted” (in the random copy  $\mathbf{c}$ ), then the decoder  $D$  never outputs a mistake.

► **Claim 4.4.** *If  $\mathbf{c}_{\bar{i}} = x_{\bar{i}}$ , then  $\Pr_D[D^w(\bar{i}) \in \{x_{\bar{i}}, \perp\}] = 1$ .*

**Proof.** By the definition of the decoder (see Figure 1), regardless of the rest of the values in the input,  $D$  always outputs either  $\mathbf{c}_{\bar{i}}$  or  $\perp$ .  $\blacktriangleleft$

The main part of the analysis takes place in the next lemma, where we assume that  $\mathbf{c}_{\bar{i}} \neq x_{\bar{i}}$  and  $\mathbf{c}$  is close to  $C(x)$ , and prove that the decoder succeeds with constant probability, as required. Recall that  $\delta_{C'}(w) < \delta_{\text{radius}}$ , where  $\delta_{\text{radius}}$  is a sufficiently small constant, to be determined later.

► **Lemma 4.5.** *Suppose that  $\mathbf{c}$  is  $4\delta_{C'}(w)$ -close to  $C(x)$  and that  $\mathbf{c}_{\bar{i}} \neq x_{\bar{i}}$ . Then,*

$$\Pr_D[D^w(\bar{i}) \in \{x_{\bar{i}}, \perp\}] = \Omega(1).$$

**Proof.** We say that a point  $\bar{u} \in [n]^d$  in the tensor code  $\mathbf{c}$  is **corrupted** if  $\mathbf{c}_{\bar{u}} \neq C(x)_{\bar{u}}$ . Since we assume that  $\mathbf{c}$  is corrupted in the point  $\bar{i}$  (which we wish to decode), by the definition of the decoder, the probability that  $D$  makes a mistake is equal to the probability that  $D$  reaches Step 4 and outputs  $\mathbf{c}_{\bar{i}}$ .

Recall that  $P_j$  is the set of points that we consider in the  $j^{\text{th}}$  iteration of the decoder. The set  $P_1$  is the singleton that contains  $\bar{i}$ ; i.e.,  $P_1 = \{\bar{i}\}$  and for every  $j \in \{2, \dots, d+1\}$  we recursively define  $P_j$  as the set of all points that reside on the  $(j-1)$ -axis-parallel lines that pass through points in  $P_{j-1}$  (see Step 3d). Note that for every  $j \in [d]$  the cardinality of  $P_j$  is equal to the number of points in a codeword of  $C_0^{\otimes j-1}$ ; that is,  $|P_j| = n^{j-1}$ . Hence, the number of points in all lines that pass through points in  $P_j$  (i.e.,  $n^j$ ) equals the number of points in a codeword of  $C_0^{\otimes j}$ . We will show that in order to corrupt  $\mathbf{c}_{\bar{i}}$  without being detected by the scPCPPs, one has to corrupt a constant fraction of a large portion of the lines that pass through points in  $P_d$ , which in turn implies that one has to corrupt a constant fraction of the tensor code  $C$ , in contradiction to our assumption that  $\delta_{C'}(w) < \delta_{\text{radius}}$ , for a sufficiently small constant  $\delta_{\text{radius}}$ .

Consider the first iteration of Step 3 (where  $j = 1$ ). Denote by  $s \triangleq i_1$  the index of the bit that we wish to decode in the line  $\mathbf{c}|_{\ell_1, \bar{i}}$ , and denote by  $b \triangleq \mathbf{c}_{\bar{i}}$  the value of  $\mathbf{c}$  at  $\bar{i}$ .

We verify that the line that passes through  $\bar{i}$  in the 1-direction is a codeword of  $C_0$  that is consistent with the value of  $\mathbf{c}$  at  $\bar{i}$ . This is done by running the verifier  $V_{s,b}$  on input  $\mathbf{c}|_{\ell_1, \bar{i}}$  and proof  $p^{1, \bar{i}}$ . Recall that the relative distance of  $C_0$  (i.e.,  $\delta(C_0)$ ) is a constant. Since  $\bar{i}$  is corrupted (i.e.,  $b = \mathbf{c}_{\bar{i}} \neq C(x)_{\bar{i}}$ ), if the line  $\mathbf{c}|_{\ell_1, \bar{i}}$  is  $\delta(C_0)/2$ -close to the line  $C(x)|_{\ell_1, \bar{i}}$  (which is a codeword of  $C_0$  that is *inconsistent* with  $\mathbf{c}_{\bar{i}}$ ), then  $\mathbf{c}|_{\ell_1, \bar{i}}$  is  $\delta(C_0)/2$ -far from any codeword  $y \in C_0$  that is *consistent* with  $\mathbf{c}_{\bar{i}}$  (i.e., such that  $y_s \neq C(x)_{\bar{i}}$ ). In this case, the verifier  $V_{s,b}$  rejects  $\mathbf{c}|_{\ell_1, \bar{i}}$  with probability at least  $\text{poly}(\delta(C_0)/2) = \Omega(1)$  (regardless of the corresponding proof), as required. Hence, in the following we assume that the line  $\mathbf{c}|_{\ell_1, \bar{i}}$  is  $\delta(C_0)/2$ -far from

$C(x)|_{\ell_{1,\bar{z}}}$ , and therefore  $P_2$  contains a constant fraction of at least  $\beta_1 \triangleq \delta(C_0)/2$  corrupted points.

We proceed by induction. Consider the  $j^{\text{th}}$  iteration, where  $2 \leq j \leq d$ . We show that if the set of points that we consider in the  $j^{\text{th}}$  iteration (the set  $P_j$ ) contains a constant fraction of corrupted points, then either the decoder rejects with constant probability in the  $j^{\text{th}}$  iteration, or  $P_{j+1}$  contains a constant fraction of corrupted points (we denote this probability by  $\beta_{j+1}$ ).

► **Claim 4.6.** *Let  $2 \leq j \leq d$  and let  $0 < \beta_j \leq 1$  be a constant. If  $P_j$  contains a at least a  $\beta_j$  fraction of corrupted points, then either:*

1. *The decoder rejects with probability at least  $\Omega(1)$  in the  $j^{\text{th}}$  iteration; or,*
2.  *$P_{j+1}$  contains at least  $\beta_{j+1} \triangleq \frac{\beta_j \cdot \delta(C_0)}{4}$  fraction of corrupted points.*

**Proof of Claim 4.6.** Consider the  $j^{\text{th}}$  iteration of Step 3. The decoder selects uniformly at random a point  $\bar{u} = (u_1, \dots, u_d) \in P_j$ . Denote by  $s = u_j$  the index of the bit that we wish to decode on the line  $\mathbf{c}|_{\ell_{j,\bar{u}}}$  (which passes through  $\bar{u}$  in the  $j^{\text{th}}$ -direction), and denote by  $b \triangleq \mathbf{c}_{\bar{u}}$  the value of  $\mathbf{c}$  at  $\bar{u}$ . By the hypothesis,  $\bar{u}$  is corrupted with probability at least  $\beta_j$ .

Next, the verifier  $V_{s,b}$  is executed on input  $\mathbf{c}|_{\ell_{j,\bar{u}}}$  and proof  $p^{j,\bar{u}}$ . Observe that if a fraction of at most  $\beta_j/2$  of the  $j$ -axis-parallel lines that pass through points in  $P_j$  (i.e.,  $\{\mathbf{c}|_{\ell_{j,\bar{z}}}\}_{\bar{z} \in P_j}$ ) are  $\delta(C_0)/2$ -far (each) from their corresponding lines in  $C(x)$ , then the decoder outputs  $\perp$  with probability at least  $\beta_j/2 \cdot \text{poly}(\delta(C_0)/2) = \Omega(1)$ , as required. This is because in this case, with probability at least  $\beta_j/2$ , we hit a line that is  $\delta(C_0)/2$ -close to its corresponding line in  $C(x)$  (but the value of this line in  $u_j$  differs from  $C(x)_{\bar{u}}$ ). As in the first iteration, this implies that this line is  $\delta(C_0)/2$ -far from any codeword  $y \in C_0$  such that  $y_s \neq C(x)_{\bar{u}}$ , and hence the verifier  $V_{s,b}$  rejects  $\mathbf{c}|_{\ell_{j,\bar{u}}}$  with probability at least  $\text{poly}(\delta(C_0)/2)$  (regardless of the corresponding proof).

Otherwise (i.e., if the above case does not hold), at least  $\beta_j/2$  of the lines in  $\{\mathbf{c}|_{\ell_{j,\bar{z}}}\}_{\bar{z} \in P_j}$  are  $\delta(C_0)/2$ -far (each) from their corresponding lines in  $C(x)$ . Therefore,  $P_{j+1}$  contains at least a  $\frac{\beta_j \cdot \delta(C_0)}{4}$  fraction of corrupted points. ◀

Note that  $P_{d+1}$  is the set of all points in  $[n]^d$ . By solving the recurrence relation, we get that  $\beta_{d+1} = \frac{\delta(C_0)^d}{2^{2d-1}}$ .<sup>16</sup> Recall that according to the hypothesis of the lemma,  $\mathbf{c}$  is  $4\delta_{\text{radius}}$ -close to  $C(x)$ . Fix the decoding radius  $\delta_{\text{radius}}$  to a sufficiently small constant such that  $4\delta_{\text{radius}} < \beta_{d+1}$ . Thus, Claim 4.6 implies that in one of the iterations the decoder must reject with probability at least  $\Omega(1)$ , as required. ◀

## Remarks

The codewords of  $C'$  are of the form  $w = (\bar{c}, \bar{p}^{\text{lines}}, \bar{p}^{\text{planes}})$ , where the three parts are of equal length. The fact that the length of each of the three parts is proportional to the others is critical. The length of  $\bar{c}$  must be proportional to the length of  $w$  in order for our code to have constant relative distance (recall that there is no guarantee on the distance of the scPCPPs). Moreover, the length of each of the scPCPP parts,  $\bar{c}$  and  $\bar{p}^{\text{lines}}$ , should be proportional to the length of  $w$  in order to obtain the average smoothness requirement (see Section 4.3).

<sup>16</sup> Recall that the fraction of corrupted points in  $P_2$  is at least  $\delta(C_0)/2$ , and that for  $2 \leq j \leq d$  the fraction of corrupted points in  $P_{j+1}$  (which we denote by  $\beta_{j+1}$ ) is at least  $\frac{\beta_j \cdot \delta(C_0)}{4}$ .



We remark that we chose our tensor code to be *systematic* only for the sake of convenience. Instead, we could have added the message itself (repeated to obtain the proper length) as a fourth part to the code  $C'$ .<sup>17</sup>

Next, we note that for the proof that our code  $C'$  is a relaxed-LDC we only use the point-line scPCPPs and ignore the plane scPCPPs (i.e., the third part of  $w$ ). Furthermore, we do not use the fact that the point-line scPCPPs are neither *strong* nor *canonical*. That is, to get only a relaxed-LDC with nearly-linear length it is enough to augment a good systematic tensor code (i.e., a tensor product of a systematic linear code with constant rate and constant relative distance) with a “regular” PCPP. However, the plane scPCPPs and the *strongness* and *canonicity* of the PCPPs will be heavily used in the proof that  $C'$  is also a strong-LTC (see Section 5).

### 4.3 Obtaining Average Smoothness

In this subsection, we conclude the proof that  $C' : \{0, 1\}^{k'} \rightarrow \{0, 1\}^{n'}$  is a relaxed-LDC. Recall that in Section 4.2 we showed a decoder  $D$  for  $C'$  (described in Figure 1) that satisfies the first two conditions of Definition 4.2, i.e., the *completeness* and (*modified*) *relaxed soundness* conditions. Next, we show that  $D$  can be modified such that it also satisfies the third and final condition of Definition 4.2, i.e., the *average smoothness* condition (which, roughly speaking, requires that the decoder makes nearly-uniform queries on average).

Denote by  $\mathcal{D}^w(i, j, r)$  the  $j^{\text{th}}$  query of the decoder  $D$  on coordinate  $i \in [k']$ , coin tosses  $r$ , and input oracle  $w$ . Recall that  $D$  satisfies the *average smoothness* condition if for every  $w \in \{0, 1\}^{n'}$  and  $v \in [n']$ , it holds that

$$\Pr_{i,j,r} [\mathcal{D}^w(i, j, r) = v] < \frac{2}{n'}, \quad (2)$$

where the probability is taken uniformly over all possible choices of  $i \in [k']$ ,  $j \in [q]$  (where  $q$  is the number of queries that  $D$  makes), and coin tosses  $r$ .

Firstly, we can relax the condition in Equation (2) and replace it with the condition

$$\Pr_{i,j,r} [\mathcal{D}^w(i, j, r) = v] = O\left(\frac{1}{n'}\right). \quad (3)$$

To see this, note that if the decoder  $D$  (which makes  $q = O(1)$  queries) satisfies Equation (3), then we can obtain a decoder  $D'$  that makes  $q' = O(q)$  queries and satisfy Equation (2) simply by running  $D$  and adding  $O(q)$  uniformly distributed “dummy” queries (whose answers the decoder ignores).

Secondly, note that by the construction of  $D$  (of Figure 1), each of the scPCPPs verifiers that are being emulated by  $D$  makes nearly-uniform queries (see Theorems 3.1 and 3.2) to the statement it refers to and to its corresponding proof. Observe that on a random index  $\bar{u} \in [k]^d$  the decoder  $D$  invokes the verifier of the *point-line* scPCPP on uniformly selected lines in a uniformly selected copy of the tensor code. Since the length of the first and second part of each codeword of  $C'$  (i.e., the tensor code and the *point-line* scPCPPs) constitutes a constant fraction of the length of each codeword of  $C'$ , the decoder  $D$  satisfies Equation (3). Finally, by the foregoing discussion,  $D$  can be modified to satisfy Equation (2).

<sup>17</sup> Actually, this approach (of adding the message itself to the output of the code) was taken in previous constructions of relaxed-LDC (see [1, 15]). By using a systematic tensor code, we circumvented this unnecessary complication.



## 5 Establishing the Strong-LTC Property

In this section we prove that the code  $C'$ , which was defined in Section 3, is a strong locally testable code.

► **Theorem 5.1.** *The code  $C' : \{0, 1\}^{k'} \rightarrow \{0, 1\}^{n'}$  as defined in Section 3 is a strong-LTC. Furthermore, it has a tester that makes nearly-uniform queries.*

In order to prove Theorem 5.1 we need to present a tester  $T$  that is given an oracle access to  $w \in \{0, 1\}^{n'}$ , makes  $O(1)$  queries to  $w$ , and satisfies the following: For all  $w \in C$  it holds that  $T^w = 1$ , and for all  $w \notin C$  it holds that  $\Pr_T[T^w = 0] \geq \text{poly}(\delta_{C'}(w))$ .

### 5.1 Outline of the Tester and its Analysis

Recall that each codeword of  $C'$  consists of three parts: (1) an alleged  $d$ -dimensional tensor code  $C = C_0^{\otimes d} : \{0, 1\}^{k^d} \rightarrow \{0, 1\}^{n^d}$ , (2) alleged scPCPPs for every 2-dimensional plane in  $C$ ; each scPCPP ascertains that the given plane is consistent with  $C$ , and (3) alleged scPCPPs for every pair of point  $\bar{i}$  in  $C$  and line  $\ell$  in  $C$  that passes through  $\bar{i}$ ; each scPCPP ascertains that a line  $\ell$  is a codeword of  $C_0$  that is consistent with the value at a point  $\bar{i}$ .

For the simplicity of the exposition, we omit the repetitions of the three parts of the code (i.e., the tensor code, the *point-line* scPCPPs, and the *plane* scPCPPs) and assume instead that the length of the each part is equal. We note that this assumption can be easily removed by using an additional consistency test. See the full details in Section 5.2.

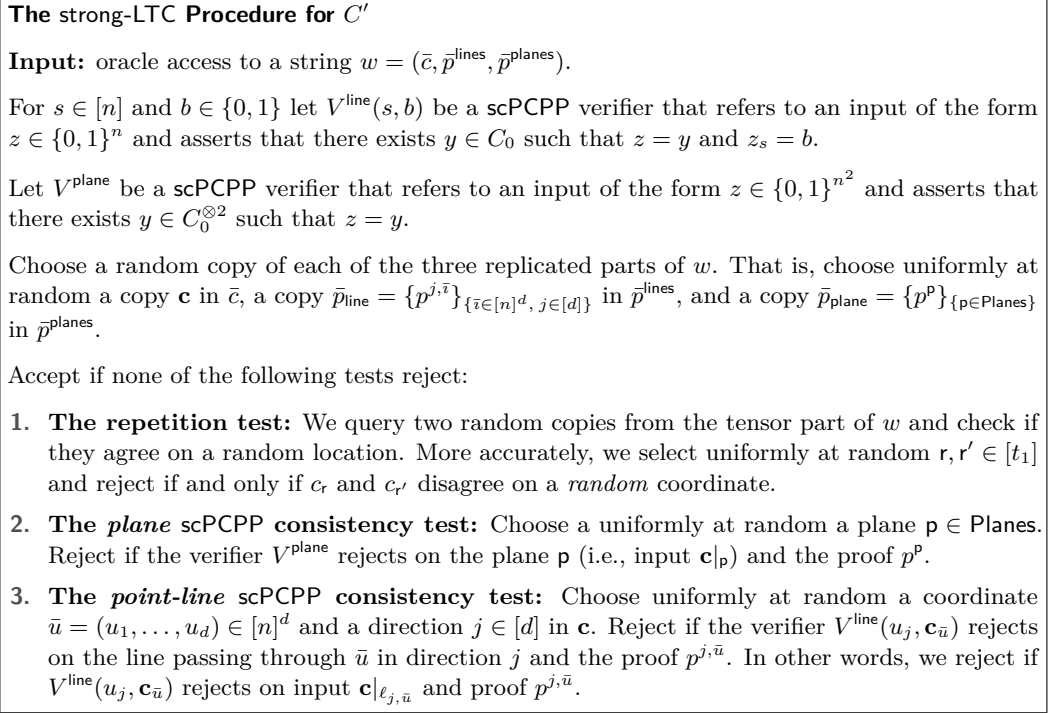
The key idea is that by the *robustness* property of tensor codes, the corruption rate of a codeword is proportional to the corruption rate of a random plane in the codeword. Hence, in order to ensure that the tensor code part of  $C'$  is valid, our tester use the *plane* scPCPPs to ascertain that a random plane is close to being valid. We note that for the tester, we do not need the *point-line* scPCPPs (which we only need for the decoder); however, since we need to ensure that also the point-line scPCPPs part is not corrupted, our tester also verifies a random *point-line* scPCPPs.

Clearly, this tester always accepts valid codewords. To analyze what happens with non-codewords consider a string that is somewhat far from  $C'$ . In this case, one of the following three cases must hold:

1. The tensor code part is far from a legal codeword of  $C^{\otimes d}$ .
2. The tensor code part is close to a legal codeword of  $C^{\otimes d}$  but the *plane* scPCPP proofs part is far from the corresponding canonical proofs.
3. The tensor code part is close to a legal codeword of  $C^{\otimes d}$  but the *point-line* scPCPP proofs part is far from the corresponding canonical proofs.

To ensure that in the first case the tester succeeds (i.e., rejects with sufficiently high probability), it is enough to test that a random plane in  $\mathbf{c}$  is close to a codeword of  $C^{\otimes 2}$ . To accomplish this, we choose uniformly at random a (2-dimensional, axis-parallel) plane and run the corresponding *plane* scPCPP verifier. This suffices, since Theorem 2.6 asserts that if a tensor  $\mathbf{c}$  is far from a legal codeword of  $C^{\otimes d}$ , then a random (2-dimensional, axis-parallel) plane in  $\mathbf{c}$  must also be far from a legal codeword of  $C^{\otimes 2}$ .

The second and third cases are similar, and so, we only sketch how to handle the second case. Assume that the tensor is close to a codeword but the *plane* scPCPPs are far from the corresponding canonical proofs. From this assumption we can deduce that there are many planes that are close to legal codewords of  $C^{\otimes 2}$ , but whose corresponding scPCPPs are far from the canonical proofs. Thus, choosing a random plane and running the corresponding



■ **Figure 2** Strong local tester for  $C'$ .

*plane* scPCPP verifier ensures that the tester rejects with a sufficiently high probability. This is due to the *strongness* and *canonicity* features of our scPCPPs.

To conclude, the tester consists of three parts: (1) a repetition test, wherein we verify the repetition structure of the tensor, (2) *plane* scPCPP consistency test, wherein we verify that a random plane in the tensor is a legal codeword; this test ensures that both the tensor code part consists of valid codewords and its *plane* scPCPPs are the corresponding canonical proofs, and (3) *point-line* scPCPP consistency test, which we perform only to verify that the *point-line* scPCPPs consists of the canonical proofs that corresponds to the tensor part of the code.

## 5.2 The Full Proof

We proceed with the full proof of Theorem 5.1, which formalizes the intuition given in the previous section. We show a strong-LTC procedure for  $C'$ . The tester  $T$  is formally described in Figure 2. Note that since both the *point-line* and *plane* scPCPP verifiers make nearly-uniform queries (and the three parts of each codeword are of equal length), then the tester  $T$  also makes nearly-uniform queries.

Consider an arbitrary input  $w \in \{0, 1\}^{n'}$  such that  $\delta_{C'}(w) \geq 0$ . We view  $w$  as a string composed of three parts as in Section 3, i.e.,  $w = (\bar{c}, \bar{p}^{\text{lines}}, \bar{p}^{\text{planes}})$ . The *completeness* of the tester is immediate: Indeed, if the input is a codeword, i.e.,  $w = C'(x)$ , then the first part of  $w$  consists of identical copies of a tensor code, and hence the codeword repetition test accepts with probability 1. Similarly, the second and third parts consists of the canonical *point-line* and *plane* scPCPP proofs for the aforementioned tensor code, respectively; hence the (one-sided error) scPCPP verifiers will accept with probability 1.

Next, we prove the *soundness* of the tester. We partition the analysis into three cases (Claim 5.2 and Lemmas 5.3 and 5.4), which we analyze in the rest of this section.

Let  $\hat{c} \in \{0, 1\}^{n^d}$  be a tensor that is closest on average to the tensors in  $\bar{c}$ , i.e., a string that minimizes  $\Delta(\bar{c}, \hat{c}^{t_1}) = \sum_{i=1}^{t_1} \Delta(c_i, \hat{c})$ . The first (and standard) claim shows that if  $\bar{c}$  is far from consisting of  $t_1$  identical tensors, then the repetition test (of Step 1) rejects with high probability. Let  $\gamma$  be a constant set to  $\delta(C)/(24d)$  (for the purpose of Lemma 5.4).

► **Claim 5.2.** *If  $\delta(\bar{c}, \hat{c}^{t_1}) \geq \frac{\gamma}{5} \cdot \delta_{C'}(w)$ , then  $\Pr_T[T^w = 0] \geq \text{poly}(\delta_{C'}(w))$ .*

**Proof.** Suppose that  $\delta(\bar{c}, \hat{c}^{t_1}) \geq \frac{\gamma}{5} \cdot \delta_{C'}(w)$ . The codeword repetition test rejects with probability at least

$$\begin{aligned} \mathbb{E}_{r, r' \in_R [t_1]} \left[ \frac{\Delta(c_r, c_{r'})}{n^d} \right] &\geq \mathbb{E}_{r \in_R [t_1]} \left[ \frac{\Delta(c_r, \hat{c})}{n^d} \right] \\ &= \frac{\Delta(\bar{c}, \hat{c}^{t_1})}{t_1 n^d}. \end{aligned}$$

Therefore,  $\Pr_T[T^w = 0] \geq \frac{\gamma}{5} \cdot \delta_{C'}(w) \geq \text{poly}(\delta_{C'}(w))$ . ◀

The following lemma shows that if  $\bar{c}$  consists of  $t_1$  nearly identical tensors that are far from a codeword of  $C$ , then due to the *robustness* feature of tensor codes, a random plane in a random copy in  $\bar{c}$  will be far from valid, and hence, Step 2 of the tester rejects with high probability.

► **Lemma 5.3.** *Assume  $\delta(\bar{c}, \hat{c}^{t_1}) < \frac{\gamma}{5} \cdot \delta_{C'}(w)$ . If  $\bar{c}$  is  $\gamma \cdot \delta_{C'}(w)$ -far from  $C^{t_1}$ , then  $\Pr_T[T^w = 0] \geq \text{poly}(\delta_{C'}(w))$ .*

**Proof.** Observe that a random copy  $\mathbf{c}$  of a tensor code in  $\bar{c}$  is  $\Omega(\delta_{C'}(w))$ -far from  $C$  with high probability. This is because  $\delta_{C^{t_1}}(\bar{c}) \leq \delta_{C^{t_1}}(\hat{c}^{t_1}) + \delta(\hat{c}^{t_1}, \bar{c})$ , which implies  $\delta_C(\bar{c}) > \frac{4\gamma}{5} \cdot \delta_{C'}(w)$ . Since at least  $2/3$  of the  $c_i$ 's are  $3 \cdot \frac{\gamma}{5} \cdot \delta_{C'}(w)$ -close to  $\hat{c}$ , these  $c_i$ 's are  $\frac{\gamma}{5} \cdot \delta_{C'}(w)$ -far from  $C$ .

Next, by the *robustness* feature of tensor codes, we deduce that if the randomly selected tensor code  $\mathbf{c}$  is  $\Omega(\delta_{C'}(w))$ -far from being valid, then a random plane of  $\mathbf{c}$  is also  $\Omega(\delta_{C'}(w))$ -far from being valid. Specifically, by Theorem 2.6, there exists a constant  $c_{\text{robust}} \in (0, 1)$  such that for every tensor  $w \in \{0, 1\}^{n^d}$  we have

$$\mathbb{E}_{\mathbf{p} \in_R \text{Planes}} [\delta(w|_{\mathbf{p}}, C^{\otimes 2})] > c_{\text{robust}} \cdot \delta_{C^{\otimes 2}}(w).$$

Hence, by an averaging argument,

$$\Pr_{\mathbf{p} \in \text{Planes}} \left[ \delta_{C^{\otimes 2}}(c|_{\mathbf{p}}) > \frac{c_{\text{robust}}}{2} \cdot \frac{\gamma}{5} \cdot \delta_{C'}(w) \right] > \frac{c_{\text{robust}}}{2} \cdot \frac{\gamma}{5} \cdot \delta_{C'}(w). \quad (4)$$

Note that, by Equation (4), with probability  $\Omega(\delta_{C'}(w))$  we select a plane that is  $\Omega(\delta_{C'}(w))$ -far from a codeword of  $C_0^{\otimes 2}$ . Given such plane, the scPCPP verifier  $V^{\text{plane}}$  rejects with probability  $\Omega(\delta_{C'}(w))$ . Thus, the tester  $T$  rejects with probability  $\text{poly}(\delta_{C'}(w))$  over the internal randomness of  $T$ . ◀

In the next lemma, we complete the analysis by assuming that  $\bar{c}$  is sufficiently close to a codeword of  $C^{t_1}$ , and showing that in this case most of the “corruption” takes place in the parts of the scPCPP proofs, and hence the scPCPP consistency tests will reject with high probability.

► **Lemma 5.4.** *If  $\bar{c}$  is  $\gamma \cdot \delta_{C'}(w)$ -close to being a codeword of  $C^{t_1}$ , then  $\Pr_T[T^w = 0] \geq \text{poly}(\delta_{C'}(w))$ .*

**Proof.** Recall that  $\gamma = \frac{\delta(C)}{24d} < \frac{\delta(C)}{2}$ . Therefore, our assumption that  $\bar{c}$  is  $\gamma \cdot \delta_{C'}(w)$ -close to being a codeword of  $C^{t_1}$  implies that there exists a *unique* codeword  $c'$  of  $C$  that minimizes the distance of  $\bar{c}' \triangleq (c')^{t_1}$  from  $\bar{c}$ . Let  $w'$  be the codeword of  $C'$  that consists of repetitions of the tensor code  $c'$  and its canonical scPCPP proofs; that is, Let  $w' = (\bar{c}', (\pi_{\text{lines}}(c'))^{t_2}, (\pi_{\text{planes}}(c'))^{t_3})$  be a codeword of  $C'$ . Denote by  $x$  the inverse of  $w'$  (i.e.,  $w' = C'(x)$ ).

It is convenient to introduce notations for the fraction of corruptions in each part of  $C'$ . Towards this end, denote the fraction of errors in the first part of the code (the copies of the tensor code) by  $\delta_{\bar{c}} = \delta(\bar{c}, \bar{c}')$ . Analogously, denote by  $\delta_{\bar{p}^{\text{lines}}}$  and  $\delta_{\bar{p}^{\text{planes}}}$  the fraction of errors in the second and third parts of  $w$  (*point-line* scPCPPs and *plane* scPCPPs), respectively. Denote by  $\delta_{\bar{p}^{\text{total}}} = (\delta_{\bar{p}^{\text{lines}}} + \delta_{\bar{p}^{\text{planes}}})/2$  the total fraction of errors in the second and third part of  $w$  together.

Observe that assuming the hypothesis of Lemma 5.4 (i.e.,  $\bar{c}$  is sufficiently close to  $\bar{c}'$ ), the scPCPPs part (i.e.,  $\bar{p}^{\text{lines}}$  and  $\bar{p}^{\text{planes}}$ ) must be somewhat far from the corresponding set of canonical scPCPP proofs; that is, assuming  $\delta_{\bar{c}} < \delta_{C'}(w)$ , then  $\delta_{\bar{p}^{\text{total}}} \geq \delta_{C'}(w)$ . Therefore, since  $\delta_{\bar{c}} \leq \gamma \cdot \delta_{C'}(w) < \delta_{C'}(w)$ , we may assume that either: (1) the *plane* scPCPPs are sufficiently corrupted, i.e.,  $\delta_{\bar{p}^{\text{planes}}} > \delta_{C'}(w)$ , or (2) the *point-line* scPCPPs are sufficiently corrupted, i.e.,  $\delta_{\bar{p}^{\text{lines}}} > \delta_{C'}(w)$ . We claim that in the first case the *plane* scPCPP consistency test will reject with high probability, and in the second case the *point-line* scPCPP consistency test will reject with high probability. We prove this in the following two claims, from which Lemma 5.4 follows.

► **Claim 5.5.** *Assuming  $\bar{c}$  is  $\gamma \cdot \delta_{C'}(w)$ -close to being a codeword of  $C^{t_1}$ , if  $\delta_{\bar{p}^{\text{planes}}} > \delta_{C'}(w)$ , then  $\Pr_T[T^w = 0] \geq \text{poly}(\delta_{C'}(w))$ .*

► **Claim 5.6.** *Assuming  $\bar{c}$  is  $\gamma \cdot \delta_{C'}(w)$ -close to being a codeword of  $C^{t_1}$ , if  $\delta_{\bar{p}^{\text{lines}}} > \delta_{C'}(w)$ , then  $\Pr_T[T^w = 0] \geq \text{poly}(\delta_{C'}(w))$ .*

Claims 5.5 and 5.6 follow immediately from the *canonicity* and *strong soundness* features of the scPCPPs (along with averaging arguments). Since the proofs of Claims 5.5 and 5.6 are similar, we conclude the proof of Lemma 5.4 by showing Claim 5.5 and defer the proof of Claim 5.6 to Section E.

**Proof of Claim 5.5.** Loosely speaking, the hypothesis of the claim guarantees that: (1)  $\bar{c}$  is close to being a *unique* codeword  $C(x)^{t_1}$ , and hence (by averaging arguments), most restrictions of a random copy  $\mathbf{c}$  in  $\bar{c} = (c_1, \dots, c_{t_1})$  to a plane cannot be significantly corrupted; (2) the *plane* scPCPPs are far, on average, from the canonical proofs that corresponds to  $C(x)$ , and thus many *plane* scPCPPs are far from the canonical proofs for the planes of  $C(x)$  they correspond to. By the foregoing, we conclude that there are many planes in  $\mathbf{c}$  that are close to planes of  $C(x)$  but their alleged *plane* scPCPP proofs are far from their canonical proofs. Thus, by the *canonicity* and *strong soundness* features of the scPCPPs, the verifier will reject with high probability. Details follow.

By the claim's hypothesis,  $\bar{c}$  is  $\delta_{\bar{c}}$ -close to  $C(x)^{t_1}$ , where  $\delta_{\bar{c}} \leq \gamma \cdot \delta_{C'}(w)$ . Hence, by an averaging argument, with probability at least  $2/3$  the random copy  $\mathbf{c}$  is  $3\delta_{\bar{c}}$ -close to  $C(x)$ . Assume from now on that this is indeed the case. We say that a point  $\bar{i} \in [n]^d$  in  $\mathbf{c}$  is *corrupted* if  $\mathbf{c}_{\bar{i}} \neq C(x)_{\bar{i}}$ , and so, there are at most  $3\delta_{\bar{c}}n^d$  corrupted points in  $\mathbf{c}$ . Since there are  $\binom{d}{2}n^{d-2}$  axis-parallel planes in  $\mathbf{c}$ , then on average, the number of corrupted points in a random axis-parallel plane in  $\mathbf{c}$  is at most  $\frac{3\delta_{\bar{c}}n^d}{\binom{d}{2}n^{d-2}} < 3\delta_{\bar{c}}n^2$ . Thus, by an averaging argument, we obtain that at most  $\frac{\delta_{\bar{p}}}{4}$  fraction of the axis-parallel planes in  $\mathbf{c}$  contain at least  $\frac{4}{\delta_{\bar{p}}} \cdot 3\delta_{\bar{c}}n^2$  corrupted points.

Secondly, we note that a random copy of the *plane* scPCPP proofs contains a fraction of  $\Omega(\delta_{C'}(w))$  corrupted points with probability  $\Omega(\delta_{C'}(w))$ . That is, by an averaging argument, with probability at least  $\delta_{\bar{p}} \triangleq \delta_{\bar{p}\text{planes}}/2$  the random copy  $\bar{p}$  in  $\bar{p}^{\text{planes}}$  is  $\delta_{\bar{p}}$ -far from its corresponding set of canonical proofs,  $\pi_{\text{planes}}(x) = \{\pi_{\text{plane}}(C(x)|_{\mathbf{p}})\}_{\mathbf{p} \in \text{Planes}}$ . Assume from now on that  $\bar{p}$  is  $\delta_{\bar{p}}$ -far from  $\pi_{\text{planes}}(x)$ . Then, by an averaging argument, we obtain that at least  $\delta_{\bar{p}}/2$  fraction of the proofs in  $\bar{p} = \{p^{\mathbf{p}}\}_{\mathbf{p} \in \text{Planes}}$  are  $\delta_{\bar{p}}/2$ -far from their corresponding (canonical) proofs  $\pi_{\text{planes}}(x)$ .

By combining the conclusions of the last two paragraphs, we deduce that  $\Omega(\delta_{C'}(w))$ -fraction of the planes  $\mathbf{p}$  in  $\mathbf{c}$  are both  $\delta(C_0^{\otimes 2})/2$ -close to the restriction of the tensor codeword  $C(x)$  to  $\mathbf{p}$ , and their corresponding proofs are  $\Omega(\delta_{C'}(w))$ -corrupted; that is, a fraction of at least  $\frac{\delta_{\bar{p}}}{4}$  of the axis-parallel planes  $\mathbf{p}$  in  $\mathbf{c}$  are  $\delta(C_0^{\otimes 2})/2$ -close to  $C(x)|_{\mathbf{p}}$  (recall that  $\frac{4}{\delta_{\bar{p}}} \cdot 3\delta_{\bar{c}} < 12\gamma < \delta(C)/2 \leq \delta(C_0^{\otimes 2})$ ), and in addition, their corresponding (alleged) *plane* scPCPP proofs in  $\{p^{\mathbf{p}}\}_{\mathbf{p} \in \text{Planes}}$  are  $\delta_{\bar{p}}/2$ -far from their (correct) canonical proofs in  $\pi_{\text{planes}}(x)$ . Denote the set of planes that satisfy the foregoing condition by BAD.

Observe that for every plane  $\mathbf{p} \in \text{BAD}$ , in order for input  $\mathbf{c}|_{\mathbf{p}}$  and proof  $p^{\mathbf{p}}$  to be a valid claim (for the input-proof language that  $V^{\text{plane}}$  verifies), one must make at least one of the following changes: (1) change a fraction of at least  $\frac{\delta_{\bar{p}}}{2}$  of the proof  $p^{\mathbf{p}}$  such that it matches  $\pi_{\text{plane}}(C(x)|_{\mathbf{p}})$ , or (2) change a fraction of at least  $\delta(C_0^{\otimes 2})/2$  of  $\mathbf{c}|_{\mathbf{p}}$  (since  $p^{\mathbf{p}}$  might be a valid proof for input  $C_0^{\otimes 2}(y) \neq \mathbf{c}|_{\mathbf{p}}$ ). Thus, for every  $\mathbf{p} \in \text{BAD}$ , the probability that  $V^{\text{plane}}$  rejects input  $\mathbf{c}|_{\mathbf{p}}$  and proof  $p^{\mathbf{p}}$  is at least polynomial in  $\delta_{C'}(w)$ .

Putting it all together, with probability  $2/3$  we hit a random copy  $\mathbf{c}$  of the tensor code that is  $3\delta_{\bar{c}}$ -close to  $C(x)$ . Furthermore, with probability at least  $\delta_{\bar{p}}$  we hit a random copy  $\bar{p}$  that is  $\delta_{\bar{p}}$ -corrupted, and subsequently, with probability  $\delta_{\bar{p}}/2$  we hit a *plane* scPCPP proof that is  $\delta_{\bar{p}}/2$ -corrupted. Finally, assuming the foregoing, the scPCPP verifier  $V^{\text{plane}}$  rejects with probability  $\text{poly}(\delta_{C'}(w))$ . Therefore,

$$\Pr_T[T^w = 0] \geq \frac{2}{3} \cdot \delta_{\bar{p}} \cdot \frac{\delta_{\bar{p}}}{2} \cdot \text{poly}(\delta_{C'}(w)) \geq \text{poly}(\delta_{C'}(w)).$$

◀

This concludes the proof of Lemma 5.4. ◀

## 6 Strong Canonical PCPs of Proximity

In this section we construct scPCPPs with *polynomial* proof length for any good linear code (see Theorem 3.1) and for any *half-space* of a any good linear code (see Theorem 3.2). Our starting point (see Corollary 6.2) is the following result of [15],<sup>18</sup> which in turn builds upon [13, Section 5.2]: For any good code  $C : \{0, 1\}^k \rightarrow \{0, 1\}^{ck}$ , there exists a strong-LTC  $C' : \{0, 1\}^k \rightarrow \{0, 1\}^{\text{poly}(k)}$  such that the first half of  $C'(x)$  consists of  $c$  blocks, each depending only on a  $k$ -bit long block of  $C(x)$ . Using this result, we construct a scPCPP for any good code  $C$ , where this construction applies the above result to several auxiliary codes that are derived from  $C$ .

### 6.1 scPCPPs for Good Codes

We start by recalling the statement of Theorem 3.1.

<sup>18</sup> Actually, Corollary 6.2 is a straightforward generalization of [15, Corollary B.3].

► **Theorem 3.1** (restated). *Let  $C : \{0, 1\}^k \rightarrow \{0, 1\}^n$  be a linear code with constant relative distance and linear length. Then, there exists a scPCPP for codewords of  $C$  (i.e., for the set  $\{C(x)\}_{x \in \{0, 1\}^k}$ ). Furthermore, the proof length of the scPCPP is  $\text{poly}(n)$ , the scPCPP verifier makes nearly-uniform queries, and the canonical scPCPP proofs are linear (over  $\text{GF}(2)$ ).*

The main technical tool upon which we rely (when proving Theorem 3.1) is the *linear inner proof systems* (hereafter, LIPS), constructed by Goldreich and Sudan. Loosely speaking, the LIPS mechanism allows to transform linear strong locally testable codes over a large alphabet into strong locally testable codes over a smaller alphabet (see [13, Section 5.2]). We encapsulate our usage of the LIPS mechanism in the following theorem, which generalizes [13, Theorem 5.20] and [13, Proposition 5.21]. Throughout this section, denote  $\mathbb{F} = \text{GF}(2)$ .

► **Theorem 6.1.** *Let  $\Sigma = \mathbb{F}^b$ . For infinitely many  $k$ , there exists  $n = \text{poly}(k)$  and a linear code  $E : \Sigma \rightarrow \mathbb{F}^n$  with constant relative distance such that the following holds. Suppose that  $C : \Sigma^K \rightarrow \Sigma^N$  is a strong-LTC that is linear over  $\mathbb{F}$  and has a (non-adaptive) tester that uses  $r$  random bits and makes nearly-uniform queries. Then, there exists  $\ell = \text{poly}(k)$  such that  $\ell$  is a multiple of  $n$ , and a linear strong-LTC  $C'' : \mathbb{F}^{bk} \rightarrow \mathbb{F}^{2^{r+1} \cdot \ell}$  such that the  $2^r \cdot \ell$ -bit long prefix of  $C''(x)$  equals  $(E(C(x)_1), \dots, E(C(x)_N))^{2^r \ell / (Nn)}$ . Moreover, the tester of  $C''$  makes nearly-uniform queries.*

As a corollary of Theorem 6.1, we obtain that any good linear code can be augmented to a linear strong-LTC with *polynomial* length, such that the prefix of the new code is closely related to that of the original code (but is *not* equal to the original code). This is done by viewing the good linear code as a trivial strong-LTC over a sufficiently large alphabet.

► **Corollary 6.2** (our starting point). *Let  $C : \{0, 1\}^k \rightarrow \{0, 1\}^{ck}$  be a good linear code with constant relative distance, where  $c \in \mathbb{N}$  is a constant. Then, for some  $M, m = \text{poly}(k)$ , there exists a linear strong-LTC  $C' : \{0, 1\}^k \rightarrow \{0, 1\}^{2M}$  and a linear code  $E : \{0, 1\}^k \rightarrow \{0, 1\}^m$ , which has constant relative distance, such that the  $M$ -bit long prefix of  $C'(x)$  equals  $(E(C(x)[1]), \dots, E(C(x)[c]))^{M/(c \cdot m)}$ , where  $C(x)[i]$  is the  $i^{\text{th}}$  block of length  $k$  in  $C(x)$ . Furthermore, the (strong) tester of  $C'$  makes nearly-uniform queries.*

We remark that Theorem 6.1 and Corollary 6.2 are straightforward generalization of [15, Theorem B.2] and [15, Corollary B.3] (respectively), and we defer their proofs to Appendix A.

## The Plan

Let  $C : \{0, 1\}^k \rightarrow \{0, 1\}^{ck}$  be a good linear code, where  $c \in \mathbb{N}$  is a constant. We construct a strong-LTC  $C'$  such that a constant fraction of each codeword  $C'(x)$  contains copies of  $C(x)$ . This, in turn, implies a scPCPP for  $C$  (see Proposition 6.5). Note that by applying Corollary 6.2 to  $C$  we obtain a strong-LTC  $C'$  such that a constant fraction of each codeword  $C'(x)$  contains copies of  $(E(C(x)[1]), \dots, E(C(x)[c]))$ , but not of  $C(x)$ . This does not seem to suffice for obtaining a scPCPP, and so we use a different approach.

We start by using Corollary 6.2 to obtain a family of linear strong-LTCs, denoted by  $\{C_i : \{0, 1\}^k \rightarrow \{0, 1\}^n\}_{i \in [ck]}$ , where  $n = \text{poly}(k)$ , with constant relative distance such that the prefix of each codeword  $C_i(x)$  contains a linear number of copies of the  $i^{\text{th}}$ -bit of  $C(x)$  (as well as other structural features that will be useful for us). This is done via the next lemma, which uses techniques from [15].

► **Lemma 6.3** (obtaining auxiliary codes  $C_i$ ). *Let  $C : \{0, 1\}^k \rightarrow \{0, 1\}^{ck}$  be a good linear code, where  $c \in \mathbb{N}$  is a constant. There exist a constant  $\alpha \in (0, 1)$ , a polynomial value  $n = \text{poly}(k)$ ,*



and a linear code  $\hat{C} : \{0, 1\}^k \rightarrow \{0, 1\}^{cn}$  with constant relative distance, which satisfy the following: For every  $i \in [ck]$ , there exists a function  $\pi_i : \{0, 1\}^k \rightarrow \{0, 1\}^{(c+1)n}$  such that the code  $C_i : \{0, 1\}^k \rightarrow \{0, 1\}^{\alpha n + cn + (c+1)n}$ , given by

$$C_i(x) = ((C(x)_i)^{\alpha n}, \hat{C}(x), \pi_i(x)),$$

is a linear strong-LTC with constant relative distance. Moreover, for every  $i \in [ck]$  the (strong) tester of  $C_i$  makes nearly-uniform queries.

We stress that the code  $\hat{C}$  (which is common to all  $C_i$ 's) is independent of  $i$  and constitutes a constant fraction of the length of each  $C_i$ .

**Proof of Lemma 6.3.** For every  $j \in [c]$ , we denote by  $C(x)[j]$  the  $j^{\text{th}}$  block of length  $k$  of  $C(x)$ . For every  $i \in [ck]$ , consider the code  $C'_i : \{0, 1\}^k \rightarrow \{0, 1\}^{(c+1)k}$  given by

$$C'_i(x) \triangleq ((C(x)_i)^k, C(x)) = ((C(x)_i)^k, C(x)[1], \dots, C(x)[c]).$$

Note that  $C'_i$  is a good linear code.

For every  $i \in [ck]$ , we apply Corollary 6.2 to  $C'_i$  and obtain a linear strong-LTC  $C''_i : \{0, 1\}^k \rightarrow \{0, 1\}^{2(c+1) \cdot n}$  with constant relative distance, which is (up to a permutation of its bit locations) of the form

$$C''_i(x) = \left( \left( E((C(x)_i)^k) \right)^t, \left( E(C(x)[1]) \right)^t, \dots, \left( E(C(x)[c]) \right)^t, \pi_i(x) \right)$$

where  $m, n = \text{poly}(k)$ , the function  $E : \{0, 1\}^k \rightarrow \{0, 1\}^m$  is a linear code with constant relative distance,  $t = n/m$ , and  $\pi_i(x) \in \{0, 1\}^{(c+1)n}$  is some string. Moreover, the (strong) tester of  $C''_i$  makes nearly-uniform queries.

Denote by  $\hat{C} : \{0, 1\}^k \rightarrow \{0, 1\}^{cn}$  the linear code (with constant relative distance) that is given by  $\hat{C}(x) = \left( \left( E(C(x)[1]) \right)^t, \dots, \left( E(C(x)[c]) \right)^t \right)$ . Since  $E$  is a linear code with constant relative distance, then  $E(0^k) = 0^m$  and  $\Delta(E(1^k), 0^m) \geq \alpha m$  for some constant  $\alpha \in (0, 1)$ . Now, for every  $i \in [ck]$ , consider the code  $C_i : \{0, 1\}^k \rightarrow \{0, 1\}^{\alpha n + cn + (c+1)n}$ , given by  $C_i(x) = ((C(x)_i)^{\alpha n}, \hat{C}(x), \pi_i(x))$ , which is obtained from  $C''_i$  by simply removing coordinates on which  $E(0^k)$  and  $E(1^k)$  agree, in each of the  $t$  copies in the first part (i.e.,  $E(C(x)_i)^k$ ).

Note that  $C_i$  has constant relative distance. Furthermore, since  $C''_i$  is linear and since we only removed coordinates on which the value is 0, the code  $C_i$  is also a linear code. Finally, by emulating the execution of the tester of  $C''_i$  on an (alleged) codeword of  $C_i$  (which can be done by returning 0 whenever a coordinate that was omitted is being queried), we obtain that  $C_i(x)$ , which is of the required form of the hypothesis, is a strong-LTC with a (strong) tester that makes nearly-uniform queries.  $\blacktriangleleft$

In the actual proof of Theorem 3.1, we will construct a code  $C'$  that encodes a message  $x$  by concatenating the encodings of  $x$  by all of the strong-LTCs in  $\{C_i : \{0, 1\}^k \rightarrow \{0, 1\}^n\}_{i \in [ck]}$  (i.e.,  $C'(x) \triangleq (C_1(x), \dots, C_{ck}(x))$ ). Thus, we will obtain a strong-LTC that (up to a permutation of the bit locations) contains copies of the entire codeword  $C(x)$  in its prefix. We remark that, in general, the concatenation of strong-LTCs is *not* a strong-LTC. However, the structure of the aforementioned family of codes (specifically, the fact that all codes in the family contains a *common* sub-code) implies that the concatenation of codes in  $\{C_i : \{0, 1\}^k \rightarrow \{0, 1\}^n\}_{i \in [ck]}$  yields a strong-LTC. The next proposition shows a sufficient condition for obtaining strong-LTCs via concatenation of strong-LTCs.

► **Proposition 6.4** (concatenating multiple encodings of strong-LTCs with a common sub-code). *Let  $C_1, \dots, C_t : \{0, 1\}^k \rightarrow \{0, 1\}^n$  be strong-LTCs with constant relative distance. Let  $I \subseteq [n]$  such that  $|I| = \Omega(n)$ , and let  $\hat{C} : \{0, 1\}^k \rightarrow \{0, 1\}^{|I|}$  be a code with constant relative distance. If  $\hat{C}(x) = C_1(x)|_I = C_2(x)|_I = \dots = C_t(x)|_I$  for every  $x \in \{0, 1\}^k$ , where  $C_i(x)|_I$  denotes the restriction of  $C_i(x)$  to  $I$ , then the code  $C'(x) \triangleq (C_1(x), \dots, C_t(x))$  is a strong-LTC with constant relative distance. Moreover, if the (strong) testers of  $C_1, \dots, C_t$  make nearly-uniform queries, then the (strong) tester of  $C'$  also makes nearly-uniform queries.*

Proposition 6.4 follows by using a tester that (1) emulates the strong-LTC tester of a randomly selected concatenated code  $C_i$  (to ascertain that each concatenated codeword is valid), and (2) tests the consistency of the common code  $\hat{C}$  in two randomly selected concatenated codes (to assure that all of the concatenated codewords encode the same message). The analysis is quite straightforward and is deferred to Appendix B.

The last tool we shall need in order to prove Theorem 3.1 is the following proposition, which allows us to transform strong-LTCs to scPCPPs for prefixes of the strong-LTCs' codewords.

► **Proposition 6.5** (from strong-LTCs to scPCPPs for related codewords). *Let  $C : \{0, 1\}^k \rightarrow \{0, 1\}^n$  be a linear code, and let  $C' : \{0, 1\}^k \rightarrow \{0, 1\}^{n'}$  be a linear strong-LTC. If there exists  $I \subseteq [n']$  where  $|I| = \Omega(n')$  and  $n' - |I| = \Omega(n')$  such that  $C'(x)|_I = (C(x))^{|I|/n}$ , then there exists a scPCPP for  $C$  (i.e., for the set of codewords  $\{C(x)\}_{x \in \{0, 1\}^k}$ ) with proof length  $O(n')$ . Moreover, the canonical scPCPP proofs are linear, and if the (strong) tester of  $C'$  makes nearly-uniform queries, then the verifier of the scPCPP for  $C$  also makes nearly-uniform queries.*

**Proof.** Let  $C$ ,  $C'$ , and  $I$  be as in the hypothesis. Assume, without loss of generality, that  $I = \{1, \dots, |I|\}$ . Denote the (strong) tester of  $C'$  by  $T$ . We use  $T$  in a black-box manner in order to construct a scPCPP for the set  $\{C(x)\}_{x \in \{0, 1\}^k}$ .

Given a codeword  $C(x)$ , the canonical scPCPP proof for  $C(x)$  is given by  $\pi(x) \triangleq C'(x)|_{[n'] \setminus I}$ , where  $C'(x)|_{[n'] \setminus I}$  is the restriction of  $C'(x)$  to the coordinates outside of  $I$ . Let  $V$  be the scPCPP verifier that gets oracle access to an alleged codeword  $w \in \{0, 1\}^n$  and oracle access to a proof oracle  $p$  of length  $n' - |I|$ . Let  $t = |I|/n$ . The verifier  $V$  emulates the execution of  $T$  on  $(w^t, p)$  as follows: Each query that  $T$  makes to the first part (which are allegedly  $C(x)^t$ ) is simulated by a corresponding query to the input oracle  $w$ ,<sup>19</sup> and each query that  $T$  makes to the other coordinates (which is allegedly  $\pi(x)$ ) is simulated by a corresponding query to the proof oracle. The verifier  $V$  accepts if and only if the emulated run of  $T$  on  $(w^t, p)$  accepted. Note that if  $T$  makes nearly-uniform queries, then  $V$  also makes nearly-uniform queries.

The *completeness* of  $V$  is immediate: If  $w$  is a codeword  $C(x)$  and  $p = \pi(x)$ , then  $(w^t, p)$  is a codeword of  $C'$ . We conclude the proof by showing the *soundness* of  $V$ . Note that  $V$  gets as input a pair of an alleged codeword  $w$  and an alleged canonical proof  $p$ . Suppose that  $\delta_{\text{PCPP}}(w, p) \triangleq \min_{x \in \{0, 1\}^n} \{ \max(\delta(x, w); \delta(\pi_{\text{canonical}}(x), p)) \} > 0$ .

For every  $x \in \{0, 1\}^n$ , either the alleged proof  $p$  is  $\delta_{\text{PCPP}}(w, p)$ -far from  $\pi_{\text{canonical}}(x)$ , or the alleged codeword is  $\delta_{\text{PCPP}}(w, p)$ -far from  $C(x)$ . In the former case, since  $|p| = n' - |I| = \Omega(n')$ , it holds that  $\delta((w^t, p), (x^t, \pi_{\text{canonical}}(x))) = \Omega(\delta_{\text{PCPP}}(w, p))$ . In the latter case, since  $\delta_{C^t}(w^t) = \delta_C(w)$  and  $|w^t| = \Omega(n')$ , it holds that  $\delta((w^t, p), (x^t, \pi_{\text{canonical}}(x))) =$

<sup>19</sup>Note that the tester expects  $t$  copies of  $C(x)$ , while the input oracle consists of a single copy. Hence, the emulation is done simply by directing the query of the  $i^{\text{th}}$  bit of the  $j^{\text{th}}$  copy to the  $i^{\text{th}}$  bit of the input oracle, for every  $i, j$ .



$\Omega(\delta_{\text{PCPP}}(w, p))$ . Therefore  $\delta_{C'}((w^t, p)) = \Omega(\delta_{\text{PCPP}}(w, p))$ , and thus the tester of  $C'$ , and subsequently the verifier  $V$ , will reject with probability  $\text{poly}(\delta_{\text{PCPP}}(w, p))$  as required.  $\blacktriangleleft$

Using Lemmas 6.3 and Propositions 6.4 and 6.5, we proceed with the proof of Theorem 3.1.

**Proof of Theorem 3.1.** Let  $c \in \mathbb{N}$  be a constant and  $C : \{0, 1\}^k \rightarrow \{0, 1\}^{ck}$  be a linear code with constant relative distance. We show a scPCPP, with polynomial proof length, for the language of all codewords of  $C$ .

First, we apply Lemma 6.3 on  $C$  and get that there exists a linear code  $\hat{C} : \{0, 1\}^k \rightarrow \{0, 1\}^{cn}$  with constant relative distance and a set of codes

$$\{C_i : \{0, 1\}^k \rightarrow \{0, 1\}^{\alpha n + cn + (c+1)n}\}_{\{i \in [ck]\}}$$

such that each  $C_i$  is a linear code with constant relative distance that is given by

$$C_i(x) = ((C(x)_i)^{\alpha n}, \hat{C}(x), \pi_i(x)),$$

where  $\alpha \in (0, 1)$ ,  $n = \text{poly}(k)$  and  $\pi_i : \{0, 1\}^k \rightarrow \{0, 1\}^{(c+1)n}$ . Moreover, each  $C_i$  makes nearly-uniform queries.

Next, we consider the code  $C'(x) \triangleq (C_1(x), \dots, C_{ck}(x))$ . Observe that, up to a permutation of the indices,  $C'$  has the form

$$C'(x) = (C(x)^{\alpha n}, \hat{C}(x)^{ck}, \pi(x)),$$

where  $\pi(x) = \pi_1(x), \dots, \pi_{ck}(x)$ . Note that  $|\hat{C}(x)^{ck}| = ck \cdot cn$ , which is a constant fraction of  $|C'(x)|$ . By Proposition 6.4, the code  $C'$  is a strong-LTC with constant relative distance that makes nearly-uniform queries.

Finally, the theorem follows by applying Proposition 6.5 to the code  $C'$  with  $I = [\alpha n \cdot ck]$ , where the code  $C$  is repeated  $\alpha n = |I|/(ck)$  times. (Indeed, we use the fact that  $|I|$  is a constant fraction of  $|C'(x)|$ .) Note that the scPCPP proof we obtain (namely,  $(\hat{C}(x)^{ck}, \pi(x))$ ) is of length  $\text{poly}(k)$ .  $\blacktriangleleft$

## 6.2 scPCPPs for Half-Spaces of Good Codes

We start by recalling the statement of Theorem 3.2.

► **Theorem 3.2 (restated).** *Let  $C : \{0, 1\}^k \rightarrow \{0, 1\}^n$  be a linear code with constant relative distance and linear length. Let  $i \in [k]$  be a location in a message and  $b \in \{0, 1\}$  a bit. Then, there exists a scPCPP for  $C_{i,b}$ , where  $C_{i,b}$  is the set of all codewords  $w$  of  $C$  such that the  $i^{\text{th}}$ -bit of  $w$  equals  $b$  (i.e.,  $w_i = b$ ). Furthermore, the proof length of the scPCPP is  $\text{poly}(n)$ , the scPCPP verifier makes nearly-uniform queries, and the scPCPP proofs are linear (over  $\text{GF}(2)$ ).*

Theorem 3.2 is obtained by using Theorem 3.1 in a black-box manner. Specifically, note that in case  $b = 0$ , the code  $C_{i,0}$  is linear, and thus we can apply Theorem 3.1 directly. On the other hand, in case  $b = 1$ , the code  $C_{i,1}$  is *not* linear, but we can “shift” it (by a fixed codeword of  $C_{i,1}$ ) and apply Theorem 3.1.

**Proof of Theorem 3.2.** In light of the above, we focus on the case in which  $b = 1$ . Assume, without loss of generality, that there exists a codeword  $c^{(i)}$  of  $C$  such that that the  $i^{\text{th}}$ -bit of  $c^{(i)}$  is 1 (otherwise, we can always reject). Consider a verifier,  $V_{i,1}$ , that gets oracle access to an input string  $w$  and a proof  $\pi$ , and proceeds as follows. The verifier  $V_{i,1}$  emulates the

execution of  $V_{i,0}$  (obtained via Theorem 3.1) on input oracle  $w + c^{(i)}$  (where the summation is point-wise over  $\mathbf{GF}(2)$ ) and its proof oracle  $\pi$  (which should be the canonical proof for  $w + c^{(i)} \in C_{i,0}$ ). Note that the verifier  $V_{i,0}$  makes nearly-uniform queries, and so  $V_{i,1}$  also makes nearly-uniform queries. We show that  $V_{i,1}$  is a scPCPP for  $C_{i,1}$ .

The *completeness* is immediate: Recall that if  $w$  is a codeword of  $C_{i,1}$ , then  $w = C(x)$  such that  $w_i = 1$ . By the linearity of  $C$ ,  $w + c^{(i)}$  is a codeword of  $C$  such that its  $i^{\text{th}}$  bit is 0 (i.e.,  $(w + c^{(i)})_i = 0$ ). Therefore, we actually invoke  $V_{i,0}$  on a codeword of  $C_{i,0}$ . For the *soundness* condition, assume that  $\delta_{C_{i,1}}(w) > 0$ . Observe that

$$\delta_{C_{i,0}}(w + c^{(i)}) = \min_{w' \in C_{i,0}} \delta(w', w + c^{(i)}) = \min_{w' \in C_{i,0}} \delta(w' + c^{(i)}, w) = \delta_{C_{i,1}}(w).$$

Therefore, the verifier  $V_{i,1}$  will reject the input  $w + c^{(i)}$  (given the corresponding canonical proof) with probability at least  $\text{poly}(\delta_{C_{i,1}}(w))$ , as required.  $\blacktriangleleft$

## 7 Application to Property Testing

In this section we give an application of our main result (Theorem 1.1) to the area of *property testing*. Specifically, we improve on the best known separation result, due to Gur and Rothblum [15], between the complexity of *decision* versus *verification* in the property testing model. Details follow.

The study of property testing, initiated by Rubinfeld and Sudan [20] and Goldreich, Goldwasser and Ron [11], considers highly-efficient randomized algorithms that solve approximate decision problems, while only inspecting a small fraction of the input. Such algorithms, commonly referred to as *testers*, are given oracle access to some object, and are required to determine whether the object has some predetermined property or is far (say, in Hamming distance) from every object that has the property.

Remarkably, it turns out that many natural properties can be tested by making relatively few queries to the object. However, there are also many natural properties that *no* tester can test efficiently. In fact, “almost all” properties require a very large query complexity to be tested. Motivated by this limitation, Gur and Rothblum [15] initiated the study of  $\mathcal{MA}$  proofs of proximity (hereafter  $\mathcal{MAP}$ s), which can be viewed as the NP proof-system analogue of property testing.

Loosely speaking, an  $\mathcal{MAP}$  is a probabilistic proof system that augments the property testing framework by allowing the tester full and free access to an (alleged) proof. That is, such a proof-aided tester for a property  $\Pi$  is given *oracle* access to an input  $x$  and *free* access to a proof string  $w$ , and should distinguish between the case that  $x \in \Pi$  and the case that  $x$  is far from  $\Pi$ , while only making a sublinear number of queries. More precisely, given a *proximity parameter*  $\varepsilon > 0$ , we require that for inputs  $x \in \Pi$ , there exist a proof that the tester accepts with high probability, and for inputs  $x$  that are  $\varepsilon$ -far from  $\Pi$  no proof will make the tester accept, except with some small probability of error. For formal definitions we refer to [15, Section 2].

As observed by [15], given an  $\mathcal{MAP}$  proof of length that is linear in the size of the object (specifically, a proof that fully describes the object), every property can be tested by only making  $O(1/\varepsilon)$  queries to the object, simply by verifying the proof’s consistency with the object. Hence, it is natural to measure the complexity of an  $\mathcal{MAP}$  by both the length of the proof and the number of queries made in order to decide whether  $x \in \Pi$  or  $\varepsilon$ -far from it. We note that a property tester can be viewed as an  $\mathcal{MAP}$  that uses a proof of length 0.

Gur and Rothblum [15] showed that the task of separating the power of property testers and  $\mathcal{MAP}$ s can be reduced to the task of designing a code that is both locally testable and

locally decodable. Furthermore, they noticed that for such a separation, *relaxed decodability* suffices. Unable to construct a code as in Theorem 1.1, Gur and Rothblum used several weaker codes to obtain partial separation results. Specifically, they proved the following theorem.

► **Theorem 7.1** (Theorems 3.1, 3.2 and 3.3 in [15]). *In all items,  $n$  denotes the length of the main input being tested.*

1. *For every constant  $\alpha > 0$ , there exists a property  $\Pi_\alpha$  that has an  $\mathcal{MAP}$  that uses a proof of length  $O(\log n)$  and makes  $\text{poly}(1/\varepsilon)$  queries for every  $\varepsilon > 1/\text{polylog}(n)$ , but for which every property tester must make  $\Omega(n^{1-\alpha})$  queries.*
2. *For every constant  $\alpha > 0$ , there exists a property  $\Pi_\alpha$  that has an  $\mathcal{MAP}$  that uses a proof of length  $O(\log n)$  and makes  $\text{poly}(\log n, 1/\varepsilon)$  queries, but for which every property tester must make  $\Omega(n^{1-\alpha})$  queries.*
3. *There exists a universal constant  $c \in (0, 1)$  and a property  $\Pi$  that has an  $\mathcal{MAP}$  that uses a proof of length  $O(\log n)$  and makes  $\text{poly}(1/\varepsilon)$  queries (without limitation on  $\varepsilon$ ), but for which every property tester must make  $n^c$  queries.*

*Furthermore, each of the above  $\mathcal{MAP}$ s has one-sided error.*

Note that each of these separation results has a drawback: The first separation works only for sufficiently large values of the proximity parameter, the second separation has non-constant query complexity for the  $\mathcal{MAP}$ s, and the third separation does not require property testers to make nearly-linear number of queries.

Plugging in the code  $C'$  from Theorem 1.1 into the framework developed by [15, Lemmas 3.4 and 3.5], we achieve the best of all the aforementioned results; that is, a separation for all values of the proximity parameter, with constant query complexity for the  $\mathcal{MAP}$ s, and nearly-linear query complexity for testers. Formally, we obtain the following separation result between  $\mathcal{MAP}$ s and property testers.

► **Theorem 1.3** (restated). *For every constant  $\alpha > 0$ , there a property  $\Pi_\alpha$  that has an  $\mathcal{MAP}$  that uses a proof of length  $O(\log n)$  and makes  $\text{poly}(1/\varepsilon)$  queries (without limitation on  $\varepsilon$ ), but for which every property tester must make  $n^{1-\alpha}$  queries. Furthermore, the  $\mathcal{MAP}$  has one-sided error.*

**Acknowledgments.** We would like to thank Or Meir for a helpful discussion regarding the robustness of tensor codes and its relation to local testability, and Michael Ben-Or for raising the issue of tolerant testing. The third author would like to thank his advisor Moni Naor for his support and encouragement.

---

## References

- 1 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.
- 2 Eli Ben-Sasson and Madhu Sudan. Robust locally testable codes and products of codes. *Random Structures & Algorithms*, 28(4):387–402, 2006.
- 3 Eli Ben-Sasson and Michael Viderman. Towards lower bounds on locally testable codes via density arguments. *Computational Complexity*, 21(2):267–309, 2012.
- 4 Benny Chor, Oded Goldreich, Eyal Kushilevitz, and Madhu Sudan. Private information retrieval. *Journal of the ACM*, 45(6):965–981, 1998.
- 5 Irit Dinur and Tali Kaufman. Dense locally testable codes cannot have constant rate and distance. In *APPROX-RANDOM*, pages 507–518, 2011.

- 6 Irit Dinur and Omer Reingold. Assignment testers: Towards a combinatorial proof of the PCP theorem. *SIAM Journal on Computing*, 36(4):975–1024, 2006.
- 7 Klim Efremenko. 3-query locally decodable codes of subexponential length. *SIAM Journal on Computing*, 41(6):1694–1703, 2012.
- 8 Katalin Friedl and Madhu Sudan. Some improvements to total degree tests. In *ISTCS*, pages 190–198, 1995.
- 9 William I. Gasarch. A survey on private information retrieval (column: Computational complexity). *Bulletin of the EATCS*, 82:72–107, 2004.
- 10 Oded Goldreich. Short locally testable codes and proofs: A survey in two parts. In *Property Testing*, pages 65–104, 2010.
- 11 Oded Goldreich, Shafi Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *Journal of the ACM*, 45(4):653–750, 1998.
- 12 Oded Goldreich and Dana Ron. On proximity oblivious testing. In *STOC*, pages 141–150, 2009.
- 13 Oded Goldreich and Madhu Sudan. Locally testable codes and PCPs of almost-linear length. *Journal of the ACM*, 53(4):558–655, 2006.
- 14 Alan Guo, Swastik Kopparty, and Madhu Sudan. New affine-invariant codes from lifting. In *ITCS*, pages 529–540. ACM, 2013.
- 15 Tom Gur and Ron D. Rothblum. Non-interactive proofs of proximity. In *ITCS*, pages 133–142. ACM, 2015.
- 16 Venkatesan Guruswami and Atri Rudra. Tolerant locally testable codes. In *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*, pages 306–317. Springer, 2005.
- 17 Jonathan Katz and Luca Trevisan. On the efficiency of local decoding procedures for error-correcting codes. In *STOC*, pages 80–86, 2000.
- 18 Tali Kaufman and Michael Viderman. Locally testable vs. locally decodable codes. In *APPROX-RANDOM*, pages 670–682, 2010.
- 19 Michal Parnas, Dana Ron, and Ronitt Rubinfeld. Tolerant property testing and distance approximation. *Journal of Computer and System Sciences*, 72(6):1012–1042, 2006.
- 20 Ronitt Rubinfeld and Madhu Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM Journal on Computing*, 25(2):252–271, 1996.
- 21 Luca Trevisan. Some applications of coding theory in computational complexity. *Electronic Colloquium on Computational Complexity (ECCC)*, 2004.
- 22 Michael Viderman. A combination of testability and decodability by tensor products. In *APPROX-RANDOM*, pages 651–662, 2012.
- 23 Michael Viderman. Strong LTCs with inverse poly-log rate and constant soundness. *Electronic Colloquium on Computational Complexity (ECCC)*, 20:22, 2013.
- 24 Sergey Yekhanin. Towards 3-query locally decodable codes of subexponential length. *Journal of the ACM*, 55(1):1, 2008.
- 25 Sergey Yekhanin. Locally decodable codes. *Foundations and Trends in Theoretical Computer Science*, 6(3):139–255, 2012.

## A Obtaining Strong LTCs from LIPS

In this appendix, we provide tools that allow us to use the *linear inner proof systems* (hereafter, LIPS), constructed by Goldreich and Sudan [13], to obtain families of **strong-LTCs** with several features that we take advantage of in Appendix 6. Specifically, we prove Theorem 6.1 and Corollary 6.2. Throughout this section, denote  $\mathbb{F} = \text{GF}(2)$ . Recall the statement of Theorem 6.1.

► **Theorem 6.1** (restated). *Let  $\Sigma = \mathbb{F}^b$ . For infinitely many  $k$ , there exists  $n = \text{poly}(k)$  and a linear code  $E : \Sigma \rightarrow \mathbb{F}^n$  such that the following holds. Suppose that  $C : \Sigma^K \rightarrow \Sigma^N$  is a strong-LTC that is linear over  $\mathbb{F}$  and has a (non-adaptive) tester that uses  $r$  random bits and makes nearly-uniform queries. Then, there exists  $\ell = \text{poly}(k)$  such that  $\ell$  is a multiple of  $n$ , and a linear strong-LTC  $C'' : \mathbb{F}^{b\ell} \rightarrow \mathbb{F}^{2^{r+1} \cdot \ell}$  such that the  $2^r \cdot \ell$ -bit long prefix of  $C''(x)$  equals  $(E(C(x)_1), \dots, E(C(x)_N))^{2^r \ell / Nn}$ . Moreover, the tester of  $C''$  makes nearly-uniform queries.*

**Proof.** We follow the proof of [13, Theorem 5.20], while using the code  $C$  of the theorem's hypothesis instead of the third ingredient in that proof. In addition, following [13, Proposition 5.21], we use composition theorems (i.e., [13, Theorem 5.15] and [13, Theorem 5.17]) that preserve the nearly-uniform distribution of the queries the verifiers (or tester) make, thus ascertaining that  $C''(x)$  has a tester that queries each location with probability  $\Theta(1/N)$ . We note that in our settings, the overhead of replacing the “vanilla” composition theorems (which are used in [13, Theorem 5.20]) with the composition theorems that preserve the nearly-uniform queries is insignificant. Details follow.

In the following description, all references refer to [13]. Recall some basics regarding the terminology used in [13]. By Definitions 5.8 and 5.9, a  $(\mathbb{F}, (q, b) \rightarrow (p, a), \delta, \gamma)$ -LIPS refers to input oracles  $X_1, \dots, X_q : [n] \rightarrow \mathbb{F}^a$  and a proof oracle  $X_{q+1} : [\ell] \rightarrow \mathbb{F}^a$ , where the input oracles provide an  $n$ -long encoding (over  $\mathbb{F}^a$ ) of a single symbol in the (much) bigger alphabet  $\mathbb{F}^b$  (i.e., this encoding is denoted  $E : \mathbb{F}^b \rightarrow (\mathbb{F}^a)^n$ ). (In addition  $\delta$  is the relative distance of the encoding used, and  $\gamma$  is the detection ratio in strong soundness. In the following, both parameters will be small constants.)

The proof of Theorem 5.20 starts with an overview (page 79), and then lists three ingredients (page 80) that will be used: (1) The Hadamard based  $(\mathbb{F}, (p_H, k_H) \rightarrow (p_H + 5, 1), 1/2, 1/8)$ -LIPS (for any choice of  $p_H$  and  $k_H$ ) of Proposition 5.18, (2) The Reed-Muller based  $(\mathbb{F}, (p_{RM}, k_{RM}) \rightarrow (p_{RM} + 4, \text{poly}(\log p_{RM} k_{RM})), 1/2, \Omega(1))$ -LIPS (for any choice of  $p_{RM}$  and  $k_{RM}$ ) of Proposition 5.18, and (3) a specific strong-LTC (namely, the strong-LTC in Part 1 of Theorem 2.4). We shall use the very same first two ingredients,<sup>20</sup> but use the code  $C$  in place of the third. Assume, without loss of generality, that the randomness complexity  $r$  of the strong (tester) of  $C$  satisfies that  $2^r$  is a multiple of  $N$ . (We remark that all three ingredients have verifiers or testers that make nearly-uniform queries, and that we compose these ingredients via the composition theorems that preserve this distribution of queries.) Specifically, the second paragraph following the ingredients-list asserts that for any desired  $p''$  and  $k''$ , an  $(\mathbb{F}, (p'', k'') \rightarrow (p'' + 13, 1), \Omega(1), \Omega(1/p'')^2)$ -LIPS with randomness  $O(p'' \log k'')$ , input length  $\text{poly}(p'' k'')$ , and proof length that are  $\text{poly}(p'' k'')$ . We shall use  $p'' = O(1)$  and  $k'' = b$ , where the  $O(1)$  stands for the query complexity of the codeword tester for  $C$ . Thus the above simplifies to asserting an  $(\mathbb{F}, (O(1), b) \rightarrow (O(1), 1), \Omega(1), \Omega(1))$ -LIPS

<sup>20</sup> We remark that while these two LIPSs are presented in [13] as if they are non-uniform, it can be verified that they can be presented in uniform terms (i.e., computable by Turing machines rather than by circuits).

with randomness  $O(\log b)$  and input/proof lengths (i.e.,  $n$  and  $\ell$ ) that are  $\text{poly}(b)$ . Without loss of generality, we may assume that  $\ell$  is a multiple of  $n$ .

Next, we wish to compose  $C$  with the above LIPS via Theorem 5.15 (instead of via Theorem 5.13, which does not preserve the nearly-uniform distribution of the queries). It follows that in Item 1 of Theorem 5.15 we use  $K, N$  and  $r$  as provided by the hypothesis and  $q = O(1)$ . For Item 2, we use  $b$  as provided by the hypothesis, ( $q = O(1)$  as above),  $p = O(1)$  and  $a = 1$ , and  $n, \ell = \text{poly}(b)$  (all fitting the LIPS above). So we have  $\Gamma = F$ , and get a strong-LTC mapping  $\mathbb{F}^{bK}$  to  $\mathbb{F}^{2^{r+1} \cdot \ell}$ , which makes nearly-uniform queries. In particular, for  $t = 2^r \ell / Nn$  (i.e.,  $tNn = 2^r \ell$ ), as shown on top of page 56 (see Equation (32)), the first half of the codewords of the resulting code have the form  $(E(C(x)_1), \dots, E(C(x)_N))^t$ , where  $x \in \mathbb{F}^{bK}$  is viewed as an element of  $\Sigma^K$ . The theorem follows.  $\blacktriangleleft$

Next, recall the statement of Corollary 6.2.

► **Corollary 6.2 (restated).** *Let  $C : \{0, 1\}^k \rightarrow \{0, 1\}^{ck}$  be a good linear code with constant relative distance, where  $c \in \mathbb{N}$  is a constant. Then, for some  $M, m = \text{poly}(k)$ , there exists a linear strong-LTC  $C' : \{0, 1\}^k \rightarrow \{0, 1\}^{2M}$  and a linear code  $E : \{0, 1\}^k \rightarrow \{0, 1\}^m$ , which has constant relative distance, such that the  $M$ -bit long prefix of  $C'(x)$  equals  $(E(C(x)[1]), \dots, E(C(x)[c]))^{M/cm}$ , where  $C(x)[i]$  is the  $i^{\text{th}}$  block of length  $k$  in  $C(x)$ . Furthermore, the (strong) tester of  $C'$  makes nearly-uniform queries.*

**Proof.** Let  $C : \mathbb{F}^k \rightarrow \mathbb{F}^{ck}$  be a good linear code. Viewing  $C$  as a mapping from  $\Sigma = \mathbb{F}^k$  to  $\Sigma^c$ , note that  $C$  is a strong-LTC, which is (trivially) checked by reading all  $c$  symbols (and hence, by definition, it makes uniform queries). The claim follows by instantiating Theorem 6.1 using the code  $C$  and taking  $b = k$ ,  $K = 1$ ,  $N = c = O(1)$ , and  $r = 0$ .  $\blacktriangleleft$

## B Concatenating Multiple Encodings of Strong LTCs

In this appendix, we show a sufficient condition for obtaining strong-LTCs via concatenation of strong-LTCs. Recall the statement of Proposition 6.4.

► **Proposition 6.4 (restated).** *Let  $C_1, \dots, C_t : \{0, 1\}^k \rightarrow \{0, 1\}^n$  be strong-LTCs with constant relative distance. Let  $I \subseteq [n]$  such that  $|I| = \Omega(n)$ , and let  $\widehat{C} : \{0, 1\}^k \rightarrow \{0, 1\}^{|I|}$  be a code with constant relative distance. If  $\widehat{C}(x) = C_1(x)|_I = C_2(x)|_I = \dots = C_t(x)|_I$  for every  $x \in \{0, 1\}^k$ , where  $C_i(x)|_I$  denotes the restriction of  $C_i(x)$  to  $I$ , then the code  $C'(x) \triangleq (C_1(x), \dots, C_t(x))$  is a strong-LTC with constant relative distance. Moreover, if the (strong) testers of  $C_1, \dots, C_t$  make nearly-uniform queries, then the (strong) tester of  $C'$  also makes nearly-uniform queries.*

**Proof.** Let  $|I| = \alpha \cdot n$  for constant  $0 \leq \alpha \leq 1$ . Assume, without loss of generality, that  $I = \{1, \dots, \alpha \cdot n\}$ . For every  $i \in [t]$ , we refer to an alleged ( $n$ -bit) codeword  $C_i(x)$  as the pair of strings  $(y_i, z_i) \in \{0, 1\}^{\alpha \cdot n} \times \{0, 1\}^{(1-\alpha) \cdot n}$ , so that  $y_i$  is the common codeword  $\widehat{C}(x)$  and  $z_i$  is the rest of the codeword.

We show a tester that, given oracle access to a binary string  $w = ((y_1, z_1), \dots, (y_t, z_t))$ , where  $(y_i, z_i) \in \{0, 1\}^n$  for every  $i \in [t]$ , accepts every codeword of  $C'$  and rejects non-codewords of  $C'$  with probability that is polynomial in their relative distance from  $C'$ . The strong-LTC procedure for  $C'$  is described in Figure 3.

Note that Step 1 of the tester  $T$  invokes the tester of a uniformly selected inner code ( $C_i$ ), and so, if the testers of  $C_1, \dots, C_t$  make nearly-uniform queries, then Step 1 of  $T$  also makes nearly-uniform queries. As for Step 2 of  $T$  (which queries a uniformly selected bit in two uniformly selected  $y_i$ 's), note that by adding two dummy queries to the second part of



**The strong-LTC Procedure for  $C'$** **Input:** a string  $((y_1, z_1), \dots, (y_t, z_t)) \in \{0, 1\}^{n \cdot t}$ .

1. **The inner strong-LTC test:** Select at random  $i \in [t]$ , and run the strong-LTC tester of  $C_i$  on  $(y_i, z_i)$ .
2. **The common codeword consistency test:** Select at random  $i_1, i_2 \in [t]$  and  $j \in [n]$ , and reject if the  $j^{\text{th}}$  bit of  $y_{i_1}$  and  $y_{i_2}$  differs.

■ **Figure 3** Strong local tester for  $C'$ .

each inner code (i.e., query a uniformly selected bit in two uniformly selected  $z_i$ 's) we ensure that the first test also makes nearly-uniform queries.

The *completeness* of the tester is straightforward. If  $((y_1, z_1), \dots, (y_t, z_t))$  is equal to  $C'(x)$  for some  $x \in \{0, 1\}^k$ , then: (1) for every  $i_1, i_2 \in [t]$  it holds that  $y_{i_1} = y_{i_2}$ , and (2) for every  $i \in [t]$  it holds that  $(y_i, z_i)$  is equal to  $C_i(x)$ . Thus the tester accepts.

Next, we show the *soundness* of the tester. Let  $w = ((y_1, z_1), \dots, (y_t, z_t))$  be  $\delta_{C'}(w)$ -far from the code  $C'$ , let  $u \in \{0, 1\}^n$  be a string that minimizes the value of  $\Delta((y_1, \dots, y_t), u^t)$ , and let  $\gamma = \delta(\widehat{C})/36$ . Suppose that  $(y_1, \dots, y_t)$  is  $\gamma \cdot \delta_{C'}(w)$ -far from  $u^t$ . In this case, the “common codeword consistency test” rejects with probability

$$\mathbb{E}_{i_1, i_2 \in [t]} \left[ \frac{\Delta(y_{i_1}, y_{i_2})}{n} \right] \geq \mathbb{E}_{i_1 \in [t]} \left[ \frac{\Delta(y_{i_1}, u)}{n} \right] = \frac{\Delta((y_1, \dots, y_t), u^t)}{n \cdot t} = \gamma \cdot \delta_{C'}(w).$$

Thus, in the sequel, we assume that  $(y_1, \dots, y_t)$  is  $\gamma \cdot \delta_{C'}(w)$ -close to  $u^t$ .

Suppose that  $u$  is  $3\gamma \cdot \delta_{C'}(w)$ -far from  $\widehat{C}$ . Since  $(y_1, \dots, y_t)$  is  $\gamma \cdot \delta_{C'}(w)$ -close to  $u^t$ , at least half of the  $y_i$ 's must be  $2\gamma \cdot \delta_{C'}(w)$ -close to  $u$ , so these  $y_i$ 's are  $\gamma \cdot \delta_{C'}(w)$ -far from  $\widehat{C}$ . Thus, in the invocation of the strong-LTC test of a random  $C_i$ , with probability  $1/2$ , the test is invoked on a string  $(y_i, z_i)$  such that  $y_i$  is  $\gamma \cdot \delta_{C'}(w)$ -far from the codewords of  $\widehat{C}$ . Since  $|I| = |y_i|$  the tester will reject with probability  $\Omega(\delta_{C'}(w))$ . Hence, in the sequel, we assume that  $u$  is  $3\gamma \cdot \delta_{C'}(w)$ -close to a codeword of  $\widehat{C}$ . Since we also assume that  $(y_1, \dots, y_t)$  is  $\gamma \cdot \delta_{C'}(w)$ -close to  $u^t$ , then by the triangle inequality, the string  $(y_1, \dots, y_t)$  is  $4\gamma \cdot \delta_{C'}(w)$ -close to a (unique, since  $4\gamma < \delta(\widehat{C})/2$ ) codeword  $\widehat{C}^t(x)$ . Furthermore, by an averaging argument, at most  $\delta_{C'}(w)/8$  fraction of the  $y_i$ 's are  $\delta(\widehat{C})/2$ -far from  $\widehat{C}(x)$ .

Since  $|\widehat{C}(x)|^t = \alpha \cdot |C'(x)|$  for a constant  $\alpha \in (0, 1)$ , and since  $(y_1, \dots, y_t)$  is  $4\gamma \cdot \delta_{C'}(w)$ -close to  $\widehat{C}^t(x)$ , then  $(z_1, \dots, z_t)$  is  $\delta_{C'}(w)/2$ -far from any  $(\hat{z}_1, \dots, \hat{z}_t) \in \{0, 1\}^{(n-|I|)t}$  such that  $(\widehat{C}^t(x), (\hat{z}_1, \dots, \hat{z}_t))$  is a codeword of  $C'$ . Thus, at least  $\delta_{C'}(w)/4$  fraction of the  $z_i$ 's are  $\delta_{C'}(w)/4$ -far from their corresponding  $\hat{z}_i$ 's. Hence, at least  $\delta_{C'}(w)/8$  fraction of the  $(y_i, z_i)$  pairs satisfy (1)  $y_i$  is  $\delta(\widehat{C})/2$ -close to  $\widehat{C}(x)$ , and (2)  $z_i$  is  $\delta_{C'}(w)/4$ -far from  $\hat{z}_i(x)$ . Therefore, if we invoke the verifier of  $C_i$  on such  $(y_i, z_i)$ , it will reject with probability  $\Omega(\delta_{C'}(w))$ . Therefore, the tester  $T$  rejects with probability  $\text{poly}(\delta_{C'}(w))$ , as required. ◀

## C Robustness of Tensor Codes

In this section we prove Theorem 2.6, which is implicit in [22]. Specifically, in [22, Theorem A.5] it is shown that for  $d \geq 3$ , if a codeword  $w$  of a  $d$ -dimensional tensor code  $C^{\otimes d}$  is corrupted, then the corruption in a random hyperplane (i.e., a  $d-1$ -dimensional subplane) of  $w$  is proportional to the corruption in the entire ( $d$ -dimensional) tensor  $w$ . By applying this theorem recursively we obtain that for *constant* values of  $d \geq 3$ , the corruption in a random

2-dimensional plane of a corrupted codeword of  $C^{\otimes d}$  is proportional to the corruption in the entire codeword. Formally, we show the following.

► **Theorem 2.6 (restated).** *Let  $C$  be a linear binary code and  $d \geq 3$  an integer. Then, there exists a constant  $c_{\text{robust}} \in (0, 1)$  such that for every tensor  $w \in \{0, 1\}^{n^d}$  it holds that*

$$\mathbb{E}_{\mathfrak{p} \in \mathcal{R}\text{Planes}} [\delta(w|_{\mathfrak{p}}, C^{\otimes 2})] > c_{\text{robust}} \cdot \delta_{C^{\otimes d}}(w).$$

We start by recalling the definition of **robustness**. Informally, we say that a tester is robust if for every word that is far from the code, the tester’s view is far in expectation from any consistent view. This notion was defined for LTCs following an analogous definition for PCPs [1].

► **Definition C.1 (Robustness).** Given a tester  $T$  for a code  $C : \{0, 1\}^k \rightarrow \{0, 1\}^n$ , for every word  $w \in \{0, 1\}^k$  we define

$$\rho^T(w) = \mathbb{E}_I [\delta(w|_I, C|_I)],$$

where  $w|_I$  denotes the local view of the tester after querying on coordinates given by  $I$ . We say that the tester  $T$  has **robustness**  $\rho_C^T$  on the code  $C$  if for every  $w \in \{0, 1\}^k$  it holds that  $\rho^T(w) \geq \rho_C^T \cdot \delta_C(w)$ .

Next, we consider the “hyperplane tester for tensor codes” of Ben-Sasson and Sudan [2]. Towards this end, we first provide a notation for hyperplanes. For every  $j \in [d]$ , and  $b \in [n]$ , we say that  $\tau$  is a  $(j, b)$ -hyperplane in  $\{0, 1\}^{n^d}$  if

$$\tau = \{(i_1, \dots, i_{j-1}, b, i_{j+1}, \dots, i_d) : \text{for all } t \in [d] \setminus \{j\} \text{ we have } i_t \in [n]\}.$$

We denote by  $\text{Hyperplanes} = \{(j, b)\text{-hyperplane}\}_{\{j \in [d], b \in [n]\}}$  the set of all hyperplanes in  $\{0, 1\}^{n^d}$ , and denote the restriction of a tensor  $w \in \{0, 1\}^{n^d}$  to a hyperplane  $\tau \in \text{Hyperplanes}$  by  $w|_{\tau} \in \{0, 1\}^{n^{d-1}}$ .

► **Definition C.2 (Hyperplane Tester for Tensor Codes).** Let  $C$  be a linear code,  $d \geq 3$  an integer, and  $w \in \{0, 1\}^{n^d}$ . The hyperplane tester for  $C^{\otimes d}$  selects uniformly at random  $\tau \in \text{Hyperplanes}$ , obtains  $w|_{\tau}$  by querying all points on  $\tau$ , and accepts if and only if  $w|_{\tau} \in C^{\otimes d-1}$ .

► **Theorem C.3 ([22, Theorem A.5]).** *Let  $C$  be a linear code and  $d \geq 3$ . Let  $T$  be the hyperplane tester for  $C^{\otimes d}$ . Then,  $\rho_{C^{\otimes d}}^T \geq \frac{\delta(C)^d}{2d^2}$ .*

We show that Theorem 2.6 follows by iterative applications of Theorem C.3.

**Proof of Theorem 2.6.** Let  $C$  be a linear code and  $d \geq 3$  a constant integer. Let  $w \in \{0, 1\}^{n^d}$  be a tensor. For every  $3 \leq t \leq d$ , let  $T_t$  be the hyperplane tester for  $C^{\otimes t}$ . Note that for every  $3 \leq t \leq d$ , the tester  $T_t$  queries a hyperplane that is allegedly a codeword of  $C^{\otimes t-1}$ ; hence  $T_{t-1}$  can be composed with  $T_t$ ; that is, we can run  $T_t$  on input  $w$ , during which  $T_t$  generates a local view  $w|_I$  to be queried, and so, we can run  $T_{t-1}$  on the local view  $w|_I$ . (Note that the composed tester  $T_3 \circ \dots \circ T_d$  queries the restriction of the input  $w$  to a uniformly selected plane  $\mathfrak{p} \in \text{Planes}$ .) The robustness of the composed tester will hence be

$$\rho_{C^{\otimes d}}^{T_3 \circ \dots \circ T_d} \geq \rho_{C^{\otimes d}}^{T_d} \cdot \rho_{C^{\otimes d-1}}^{T_{d-1}} \cdot \dots \cdot \rho_{C^{\otimes 3}}^{T_3}.$$

By Theorem C.3, for every  $t \geq 3$  we have  $\rho_{C^{\otimes t}}^{T_t} \geq \frac{\delta(C)^t}{2t^2}$ . Thus, for constant  $d \geq 3$  it holds that  $c_{\text{robust}} \triangleq \rho_{C^{\otimes d}}^{T_3 \circ \dots \circ T_d}$  is a positive constant that depends only on  $\delta(C)$  and  $d$ . ◀



## D Average Smoothness and Error Reduction for Relaxed LDCs

In this appendix, following [1, Section 4.2], we show that the modified definition of relaxed-LDCs (see Definition 4.2) implies the standard definition of relaxed-LDCs (see Definition 2.2). Towards this end we need to show the following: (1) The soundness can be increased from  $\Omega(1)$  (as in Condition 2 of Definition 4.2) to  $2/3$  (as in Condition 2 of Definition 2.2), and (2) the *average smoothness* (i.e., Condition 3 of Definition 4.2) can be replaced with the *success rate* condition (i.e., Condition 3 of Definition 2.2). Both claims were shown in [1]; we provide their proofs (adapted to our settings) for completeness.

We start by showing how to perform error-reduction for relaxed-LDC with soundness  $\Omega(1)$ . Recall that the decoder is required to successfully decode each valid codeword, and in addition, given a somewhat corrupted codeword the decoder is required to either decode successfully or abort with probability  $\Omega(1)$ . On the face of it, it may seem that standard error reduction cannot be applied (since we start with a large error probability). However, the error reduction can be simply performed by repeating the execution of the decoder, outputting a bit only if all invocations returned this bit, and aborting otherwise. We remark that the above may cause an increase in the number of indices on which the decoder aborts (with probability at least  $2/3$ ). However, in the modified definition (i.e., Definition 4.2) there is no restriction on the success rate.

► **Proposition D.1.** *Let  $C : \{0, 1\}^k \rightarrow \{0, 1\}^n$  be a modified relaxed-LDC, according to Definition 4.2. Then,  $C$  has a modified relaxed-LDC decoder that also satisfies Condition 2 of Definition 2.2.*

**Proof.** Let  $C$  be a modified relaxed-LDC. Denote its decoder by  $D$ . There exists a constant  $p > 0$  such that for every string  $w$  that is sufficiently close to a codeword of  $C$  it holds that  $\Pr_D[D^w(i) = \{x_i, \perp\}] \geq p$ . Consider a decoder  $D'$  that operates follows:  $D'$  executes the original decoder  $D$  (with fresh randomness) for  $r$  times, where  $r$  is a constant to be determined later. If all of the executions are consistent, i.e., there exists an  $a \in \{0, 1, \perp\}$  such that in every execution  $D^w(i) = a$ , then  $D'$  output  $a$ ; otherwise,  $D'$  output  $\perp$ . (We remark that the distribution of queries of  $D'$  is identical to that of  $D$ , and thus  $D'$  also satisfies the *average smoothness* condition.)

Note that the new decoder  $D'$  satisfies Condition 1 of Definition 2.2 (the *completeness* condition). Moreover,  $D'$  satisfies Condition 2 of Definition 2.2: Indeed, given  $w$  that is sufficiently close to  $C(x)$ , the probability that  $D'$  errs is at most  $p' = (1 - p)^r$ . Hence, by fixing  $r = 2/p$  we get that  $\Pr_{D'}[D'^w(i) = \{x_i, \perp\}] \geq 1 - p' \geq 2/3$ , as needed. ◀

Finally, we show that the *average smoothness* condition (i.e., Condition 3 of Definition 4.2) can be replaced by the *success rate* condition (i.e., Condition 3 of Definition 2.2, which limits the number of indices upon which the decoder aborts (with probability at least  $2/3$ )). The key idea is that a decoder that satisfies the completeness and soundness conditions (i.e., Conditions 1 and 2 of Definition 2.2) only aborts if the local view of the codeword that it queries contains a corrupted point. By the *average smoothness*, on average the decoder will only query a corrupted point with low probability. Thus, by an averaging argument, we can deduce that there is a small number of indices upon which the decoder might abort.

► **Proposition D.2.** *Let  $C : \{0, 1\}^k \rightarrow \{0, 1\}^n$  be a linear code, and let  $D$  be a constant-query decoder for  $C$  that satisfies Conditions 1 and 2 of Definition 2.2 as well as Condition 3 of Definition 4.2 (i.e., *average smoothness*). Then,  $C$  satisfies all three conditions of Definition 2.2.*

**Proof.** Let the code  $C$  and the decoder  $D$  be as in the hypothesis of the proposition. Denote the (constant) query complexity of  $D$  by  $q$ . According to Condition 1, for any  $x \in \{0, 1\}^k$  and every  $i \in [k]$ , it holds that  $\Pr[D^{C(x)}(i) = x_i] = 1$ . Considering any  $w$  that is  $\delta$ -close to  $C(x)$  (where  $\delta \leq \delta_{\text{radius}}$ ), the probability that given a *uniformly distributed* index  $i \in [k]$  the decoder  $D$  queries a location on which  $w$  and  $C(x)$  disagree is at most  $q \cdot (2/n) \cdot \delta n = 2q\delta$ . This is due to the fact that, for a uniformly distributed  $i$ , no position is queried with probability greater than  $2/n$ .

Let  $p_i^w$  denote the probability that on input  $i$  the decoder  $D$  queries a location on which  $w$  and  $C(x)$  disagree. We have just established that  $(1/k) \cdot \sum_{i=1}^k p_i^w \leq 2q\delta$ . By an averaging argument, for  $I_w \triangleq \{i \in [k] : p_i^w \leq 1/3\}$ , it holds that  $|I_w| \geq (1 - 6q\delta) \cdot k$ . Observe that for any  $i \in I_w$ , it holds that  $\Pr[D^w(i) = x_i] \geq 1 - 1/3 = 2/3$ , as required.  $\blacktriangleleft$

## E Proof of Claim 5.6

In this section we provide the proof of Claim 5.6. The proof is similar to the proof of Claim 5.5. However, note that Claims 5.5 and 5.6 deal with different objects: While Claim 5.5 deals with the planes of the tensor code and the *plane* scPCPPs, Claim 5.6 deals with the lines of the tensor and the *point-line* scPCPPs. In particular, every plane in the tensor code is coupled with a *unique* plane scPCPP proof, whereas every line in the tensor code is coupled with  $n$  *different* point-line scPCPPs, one for each point on the line. We begin by restating Claim 5.6. Recall that  $\gamma = \delta(C)/(24d)$ .

**► Claim 5.6 (restated).** *Assuming  $\bar{c}$  is  $\gamma \cdot \delta_{C'}(w)$ -close to being a codeword of  $C^{t_1}$ , if  $\delta_{\bar{p}^{\text{lines}}} > \delta_{C'}(w)$ , then  $\Pr_T[T^w = 0] \geq \text{poly}(\delta_{C'}(w))$ .*

**Proof.** By the lemma's hypothesis,  $\bar{c}$  is  $\delta_{\bar{c}}$ -close to  $C(x)^{t_1}$ , where  $\delta_{\bar{c}} \leq \gamma \cdot \delta_{C'}(w)$ . By an averaging argument, with probability at least  $2/3$  the random copy  $\mathbf{c}$  is  $3\delta_{\bar{c}}$ -close to  $C(x)$ . We say that a point  $\bar{i} \in [n]^d$  in  $\mathbf{c}$  is *corrupted* if  $\mathbf{c}_{\bar{i}} \neq C(x)_{\bar{i}}$  and so, there are at most  $3\delta_{\bar{c}}n^d$  corrupted points in  $\mathbf{c}$ . Since there are  $d \cdot n^{d-1}$  axis-parallel lines in  $\mathbf{c}$ , then on average, the number of corrupted points in a random axis-parallel line is at most  $\frac{3\delta_{\bar{c}}n^d}{d \cdot n^{d-1}} \leq 3\delta_{\bar{c}}n$ . Thus, by an averaging argument, we obtain that at most  $\frac{\delta_{\bar{p}}}{4}$  fraction of the axis-parallel lines in  $\mathbf{c}$  contain at least  $\frac{4}{\delta_{\bar{p}}} \cdot 3\delta_{\bar{c}}n$  corrupted points.

Recall that every axis-parallel line  $\ell$  has  $n$  corresponding *point-line* scPCPP proofs (one for each point on  $\ell$ ). For every line  $\ell$  we view these  $n$  proofs as one concatenated proof for the line  $\ell$ . By an averaging argument, with probability at least  $\delta_{\bar{p}} \triangleq \delta_{\bar{p}^{\text{lines}}}/2$  the random copy  $\bar{p}$  in  $\bar{p}^{\text{lines}}$  is  $\delta_{\bar{p}}$ -far from its corresponding set of canonical proofs,  $\pi_{\text{lines}}(x)$ . Assume from now on that  $\bar{p}$  is  $\delta_{\bar{p}}$ -far from  $\pi_{\text{lines}}(x)$ . By another averaging argument, at least a  $\delta_{\bar{p}}/2$  fraction of the concatenated line proofs (i.e., proofs which consists of  $n$  point-line scPCPP proofs) are  $\delta_{\bar{p}}/2$ -far from their corresponding (concatenated) canonical line proofs.

By combining the conclusions of the last two paragraphs, we deduce that  $\Omega(\delta_{C'}(w))$ -fraction of the axis-parallel lines  $\ell$  in  $\mathbf{c}$  are both  $\delta(C_0)/2$ -close to the restriction of the tensor codeword  $C(x)$  to  $\ell$ , and their corresponding (concatenated) proofs are  $\Omega(\delta_{C'}(w))$ -corrupted; that is, there is a subset of lines, denoted BAD, which consists of at least  $\frac{\delta_{\bar{p}}}{4}$  fraction of all the lines in  $\mathbf{c}$  that are  $\delta(C_0)/2$ -close to  $C(x)|_{\ell}$  (recall that  $\delta_{\bar{c}} \leq \gamma \cdot \delta_{C'}(w)$  and  $\delta_{\bar{p}} > \delta_{C'}(w)$ ), therefore  $\frac{12 \cdot \delta_{\bar{c}}}{\delta_{\bar{p}}} < \delta(C_0)/2$ , and in addition satisfy the following: For every  $\ell \in \text{BAD}$ , the  $n$  (alleged) *point-line* scPCPP proofs that correspond to  $\ell$  are  $\delta_{\bar{p}}/2$ -far from their (correct) canonical proofs in  $\pi_{\text{lines}}(x)$ . By an averaging argument, for every  $\ell \in \text{BAD}$  it holds that  $\delta_{\bar{p}}/4$  fraction of the point-line PCPP proofs that correspond to the line  $\ell$  (recall that there are  $n$  such proofs) are  $\delta_{\bar{p}}/4$ -far from their canonical proof in  $\pi_{\text{lines}}(x)$ .

Recall that the tester chooses a line  $\ell = \ell_{j,\bar{i}}$  by sampling uniformly at random a point  $\bar{i} \in [n]^d$  and a direction  $j \in [d]$ . Notice that for if  $\ell \in \text{BAD}$ , then with probability  $\delta_{\bar{p}}/4$ , in order for input  $\mathbf{c}|_{\ell}$  and the proof  $p^{\ell_{j,\bar{i}}}$  (that refers to the same line as  $\ell$ ) to be a valid claim for the input-proof language that  $V^{\text{line}}(i_j, \mathbf{c}_{\bar{i}})$  verifies, one must make at least one of the following changes: (1) change a fraction of at least  $\frac{\delta_{\bar{p}}}{4}$  of the proof  $p^{\ell_{j,\bar{i}}}$  such that it matches  $\pi_{\text{line}}(C(x)|_{\ell_{j,\bar{i}}}, i_j)$ , or (2) change a fraction of at least  $\delta(C_0)/2$  of  $\mathbf{c}|_{\ell}$  (since  $p^{\ell_{j,\bar{i}}}$  might be a valid proof for input  $C_0(y) \neq \mathbf{c}|_{\ell}$ ). Thus, for every  $\ell_{j,\bar{i}} \in \text{BAD}$ , the probability that  $V^{\text{line}}(i_j, \mathbf{c}_{\bar{i}})$  rejects input  $\mathbf{c}|_{\ell_{j,\bar{i}}}$  and proof  $p^{\ell_{j,\bar{i}}}$  is at least polynomial in  $\delta_{C'}(w)$ .

Putting it all together, with probability  $2/3$  we hit a random copy  $\mathbf{c}$  of the tensor code that is  $3\delta_{\bar{c}}$ -close to  $C(x)$ . Furthermore, with probability at least  $\delta_{\bar{p}}$  we hit a random copy  $\bar{p}$  that is  $\delta_{\bar{p}}$ -corrupted, and subsequently, with probability  $\delta_{\bar{p}}/2$  we hit a set of  $n$  line scPCPP proofs that are  $\delta_{\bar{p}}/2$ -corrupted. Moreover, with probability at least  $\delta_{\bar{p}}/4$  we hit a point-line scPCPP proof that is  $\delta_{\bar{p}}/4$  corrupted. Finally, assuming the foregoing, the corresponding scPCPP verifier rejects with probability  $\text{poly}(\delta_{C'}(w))$ . Therefore,

$$\Pr_T[T^w = 0] \geq \frac{2}{3} \cdot \delta_{\bar{p}} \cdot \frac{\delta_{\bar{p}}}{2} \cdot \frac{\delta_{\bar{p}}}{4} \cdot \text{poly}(\delta_{C'}(w)) \geq \text{poly}(\delta_{C'}(w)).$$

◀

# An Entropy Sumset Inequality and Polynomially Fast Convergence to Shannon Capacity Over All Alphabets

Venkatesan Guruswami\* and Ameya Velingker†

Computer Science Department, Carnegie Mellon University  
Pittsburgh, PA, USA  
guruswami@cmu.edu, avelingk@cs.cmu.edu

---

## Abstract

We prove a lower estimate on the increase in entropy when two copies of a conditional random variable  $X|Y$ , with  $X$  supported on  $\mathbb{Z}_q = \{0, 1, \dots, q-1\}$  for prime  $q$ , are summed modulo  $q$ . Specifically, given two i.i.d. copies  $(X_1, Y_1)$  and  $(X_2, Y_2)$  of a pair of random variables  $(X, Y)$ , with  $X$  taking values in  $\mathbb{Z}_q$ , we show

$$H(X_1 + X_2 | Y_1, Y_2) - H(X|Y) \geq \alpha(q) \cdot H(X|Y)(1 - H(X|Y))$$

for some  $\alpha(q) > 0$ , where  $H(\cdot)$  is the normalized (by factor  $\log_2 q$ ) entropy. In particular, if  $X|Y$  is not close to being fully random or fully deterministic and  $H(X|Y) \in (\gamma, 1-\gamma)$ , then the entropy of the sum increases by  $\Omega_q(\gamma)$ . Our motivation is an effective analysis of the finite-length behavior of polar codes, for which the linear dependence on  $\gamma$  is quantitatively important. The assumption of  $q$  being prime is necessary: for  $X$  supported uniformly on a proper subgroup of  $\mathbb{Z}_q$  we have  $H(X+X) = H(X)$ . For  $X$  supported on infinite groups without a finite subgroup (the torsion-free case) and no conditioning, a sumset inequality for the absolute increase in (unnormalized) entropy was shown by Tao in [20].

We use our sumset inequality to analyze Arıkan’s construction of polar codes and prove that for any  $q$ -ary source  $X$ , where  $q$  is any fixed prime, and any  $\epsilon > 0$ , polar codes allow *efficient* data compression of  $N$  i.i.d. copies of  $X$  into  $(H(X) + \epsilon)N$   $q$ -ary symbols, *as soon as  $N$  is polynomially large in  $1/\epsilon$* . We can get capacity-achieving source codes with similar guarantees for composite alphabets, by factoring  $q$  into primes and combining different polar codes for each prime in factorization.

A consequence of our result for noisy channel coding is that for *all* discrete memoryless channels, there are explicit codes enabling reliable communication within  $\epsilon > 0$  of the symmetric Shannon capacity for a block length and decoding complexity bounded by a polynomial in  $1/\epsilon$ . The result was previously shown for the special case of binary-input channels [7, 9], and this work extends the result to channels over any alphabet.

**1998 ACM Subject Classification** E.4 Coding and Information Theory

**Keywords and phrases** Polar codes, polynomial gap to capacity, entropy sumset inequality, arbitrary alphabets

**Digital Object Identifier** 10.4230/LIPIcs.CCC.2015.42

---

\* Part of this work was done while visiting Microsoft Research New England. Research supported in part by NSF grants CCF-0963975 and CCF-1422045.

† Part of this work was done while visiting Microsoft Research New England. Research supported in part by NSF grant CCF-0963975.



## 1 Introduction

In a remarkable work, Arikan [2] introduced the technique of channel polarization, and used it to construct a family of binary linear codes called polar codes that achieve the symmetric Shannon capacity of binary-input discrete memoryless channels in the limit of large block lengths. Polar codes are based on an elegant recursive construction and analysis guided by information-theoretic intuition. Arikan’s work gave a construction of binary codes, and this was subsequently extended to general alphabets in [18]. In addition to being an approach to realize Shannon capacity that is radically different from prior ones, channel polarization turns out to be a powerful and versatile primitive applicable in many other important information-theoretic scenarios. For instance, variants of the polar coding approach give solutions to the lossless and lossy source coding problem [3, 13], capacity of wiretap channels [15], the Slepian-Wolf, Wyner-Ziv, and Gelfand-Pinsker problems [11], coding for broadcast channels [5], multiple access channels [19, 1], interference networks [21], etc. We recommend the well-written survey by Şaşoğlu [17] for a detailed introduction to polar codes.

The advantage of polar codes over previous capacity-achieving methods (such as Forney’s concatenated codes that provably achieved capacity) was highlighted in a recent work [7] where *polynomial convergence to capacity* was shown in the *binary* case (this was also shown independently in [9]). Specifically, it was shown that polar codes enable approaching the symmetric capacity of binary-input memoryless channels within an additive gap of  $\epsilon$  with block length, construction, and encoding/decoding complexity all bounded by a polynomially growing function of  $1/\epsilon$ . Polar codes are the first and currently only known construction which provably have this property, thus providing a formal complexity-theoretic sense in which they are the first constructive capacity-achieving codes.

The main objective of this paper is to extend this result to the non-binary case, and we manage to do this for *all* alphabets in this work. We stress that the best previously proven complexity bound for communicating at rates within  $\epsilon$  of capacity of channels with non-binary inputs was *exponential* in  $1/\epsilon$ . Our work shows the polynomial solvability of the central computational challenge raised by Shannon’s non-constructive coding theorems, in the full generality of *all discrete sources* (for compression/noiseless coding) and *all discrete memoryless channels* (for noisy coding).

The high level approach to prove the polynomially fast convergence to capacity is similar to what was done in [7], which is to replace the appeal to general martingale convergence theorems (which lead to ineffective bounds) with a more direct analysis of the convergence rate of a specific martingale of entropies.<sup>1</sup> However, the extension to the non-binary case is far from immediate, and we need to establish a quantitatively strong “entropy increase lemma” (see details in Section 4) over all prime alphabets. The corresponding inequality admits an easier proof in the binary case, but requires more work for general prime alphabets. For alphabets of size  $m$  where  $m$  is not a prime, we can construct a capacity-achieving code by combining together polar codes for each prime dividing  $m$ .

In the next section, we briefly sketch the high level structure of polar codes, and the crucial role played by a certain “entropy sumset inequality” in our effective analysis. Proving this entropic inequality is the main new component in this work, though additional technical work is needed to glue it together with several other ingredients to yield the overall coding result.

---

<sup>1</sup> The approach taken in [9] to analyze the speed of polarization for the binary was different, based on channel Bhattacharyya parameters instead of entropies. This approach does not seem as flexible as the entropic one to generalize to larger alphabets.

## 2 Overview of the Contribution

In order to illustrate our main contribution, which is an inequality on conditional entropies for inputs from prime alphabets, in a simple setting, we will focus on the source coding (lossless compression) model in this paper. The consequence of our results for channel coding, which is not immediate but follows in a standard manner from compression of sources with side information (see for instance [17, Sec 2.4]), is stated in Theorem 2.3.

Let  $\mathbb{Z}_q = \{0, 1, \dots, q-1\}$  denote the additive group of integers modulo  $q$ . Suppose  $X$  is a source (random variable) over  $\mathbb{Z}_q$  (with  $q$  prime), with entropy  $H(X)$  (throughout the paper, by entropy we will mean the entropy normalized by a  $\lg q$  factor, so that  $H(X) \in [0, 1]$ ). The source coding problem consists of compressing  $N$  i.i.d. copies  $X_0, X_1, \dots, X_{N-1}$  of  $X$  to  $\approx H(X)N$  (say  $(H(X) + \epsilon)N$ ) symbols from  $\mathbb{Z}_q$ . The approach based on channel polarization is to find an explicit permutation matrix  $A \in \mathbb{Z}_q^{N \times N}$ , such that if  $(U_0, \dots, U_{N-1})^t = A(X_0, \dots, X_{N-1})^t$ , then in the limit of  $N \rightarrow \infty$ , for most indices  $i$ , the conditional entropy  $H(U_i|U_0, \dots, U_{i-1})$  is either  $\approx 0$  or  $\approx 1$ . Note that the conditional entropies at the source  $H(X_i|X_0, \dots, X_{i-1})$  are all equal to  $H(X)$  (as the samples are i.i.d.). However, after the linear transformation by  $A$ , the conditional entropies get *polarized* to the boundaries 0 and 1. By the chain rule and conservation of entropy, the fraction of  $i$  for which  $H(U_i|U_0, \dots, U_{i-1}) \approx 1$  (resp.  $\approx 0$ ) must be  $\approx H(X)$  (resp.  $\approx 1 - H(X)$ ).

The polarization phenomenon is used to compress the  $X_i$ 's as follows: The encoder only outputs  $U_i$  for indices  $i \in B$  where  $B = \{i \mid H(U_i|U_0, \dots, U_{i-1}) > \zeta\}$  for some tiny  $\zeta = \zeta(N) \rightarrow 0$ . The decoder (decompression algorithm), called a *successive cancellation decoder*, estimates the  $U_i$ 's in the order  $i = 0, 1, \dots, N-1$ . For indices  $i \in B$  that are output at the encoder, this is trivial, and for other positions, the decoder computes the maximum likelihood estimate  $\hat{u}_i$  of  $U_i$ , assuming  $U_0, \dots, U_{i-1}$  equal  $\hat{u}_0, \dots, \hat{u}_{i-1}$ , respectively. Finally, the decoder estimates the inputs at the source by applying the inverse transformation  $A^{-1}$  to  $(\hat{u}_0, \dots, \hat{u}_{N-1})^t$ .

The probability of incorrect decompression (over the randomness of the source) is upper bounded, via a union bound over indices outside  $B$ , by  $\sum_{i \notin B} H(U_i|U_0, \dots, U_{i-1}) \leq \zeta N$ . Thus, if  $\zeta \ll 1/N$ , we have a reliable lossless compression scheme. Thus, in order to achieve compression rate  $H(X) + \epsilon$ , we need a polarizing map  $A$  for which  $H(U_i|U_0, \dots, U_{i-1}) \ll 1/N$  for at least  $1 - H(X) - \epsilon$  fraction of indices. This in particular means that  $H(U_i|U_0, \dots, U_{i-1}) \approx 0$  or  $\approx 1$  for all but a vanishing fraction of indices, which can be compactly expressed as  $\mathbf{E}_i[H(U_i|U_0, \dots, U_{i-1})(1 - H(U_i|U_0, \dots, U_{i-1}))] \rightarrow 0$  as  $n \rightarrow \infty$ .

Such polarizing maps  $A$  are in fact implied by a source coding solution, and exist in abundance (a random invertible map works w.h.p.). The big novelty in Arıkan's work is an explicit recursive construction of polarizing maps, which further, due to their recursive structure, enable efficient maximum likelihood estimation of  $U_i$  given knowledge of  $U_0, \dots, U_{i-1}$ .

Arıkan's construction is based on recursive application of the basic  $2 \times 2$  invertible map  $K = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$ .<sup>2</sup> While Arıkan's original analysis was for the binary case, the same construction based on the matrix  $K$  also works for any prime alphabet [18]. Let  $A_n$  denote the matrix of the polarizing map for  $N = 2^n$ . In the base case  $n = 1$ , the outputs are  $U_0 = X_0 + X_1$  and  $U_1 = X_1$ . If  $X_0, X_1 \sim X$  are i.i.d., the entropy  $H(U_0) = H(X_0 + X_1) > H(X)$  (unless  $H(X) \in \{0, 1\}$ ), and by the chain rule  $H(U_1|U_0) < H(X)$ , thereby creating a small separation in the entropies. Recursively, if  $(V_0, \dots, V_{2^{n-1}-1})$  and  $(T_0, \dots, T_{2^{n-1}-1})$  are the outputs of

<sup>2</sup> Subsequent work established that polarization is a common phenomenon that holds for most choices of the "base" matrix instead of just  $K$  [12].



$A_{n-1}$  on the first half and second half of  $(X_0, \dots, X_{2^n-1})$ , respectively, then the output  $(U_0, \dots, U_{2^n-1})$  satisfies  $U_{2i} = V_i + T_i$  and  $U_{2i+1} = T_i$ . If  $H_n$  denotes the random variable equal to  $H(U_i|U_0, \dots, U_{i-1})$  for a random  $i \in \{0, 1, \dots, 2^n-1\}$ , then the sequence  $\{H_n\}$  forms a bounded martingale. The polarization property, namely that  $H_n \rightarrow \text{Bernoulli}(H(X))$  in the limit of  $n \rightarrow \infty$ , can be shown by appealing to the martingale convergence theorem. However, in order to obtain a finite upper bound on  $n(\epsilon)$ , the value of  $n$  needed for  $\mathbf{E}[H_n(1 - H_n)] \leq \epsilon$  (so that most conditional entropies to polarize to  $< \epsilon$  or  $> 1 - \epsilon$ ), we need a more quantitative analysis. This was done for the binary case in [7], by quantifying the increase in entropy  $H(V_i + T_i|V_0, \dots, V_{i-1}, T_0, \dots, T_{i-1}) - H(V_i|V_0, \dots, V_{i-1})$  at each stage, and proving that the entropies diverge apart at a sufficient pace for  $H_n$  to polarize to 0/1 exponentially fast in  $n$ , namely  $\mathbf{E}[H_n(1 - H_n)] \leq \rho^n$  for some absolute constant  $\rho < 1$ .

The main technical challenge in this work is to show an analogous entropy increase lemma for all prime alphabets. The primality assumption is necessary, because a random variable  $X$  uniformly supported on a proper subgroup has  $H(X) \notin \{0, 1\}$  and yet  $H(X + X) = H(X)$ . Formally, we prove:

► **Theorem 2.1.** *Let  $(X_i, Y_i)$ ,  $i = 1, 2$  be i.i.d. copies of a correlated random variable  $(X, Y)$  with  $X$  supported on  $\mathbb{Z}_q$  for a prime  $q$ . Then for some  $\alpha(q) > 0$ ,*

$$H(X_1 + X_2|Y_1, Y_2) - H(X|Y) \geq \alpha(q) \cdot H(X|Y)(1 - H(X|Y)). \quad (1)$$

The *linear* dependence of the entropy increase on the quantity  $H(X|Y)(1 - H(X|Y))$  is crucial to establish a speed of polarization adequate for polynomial convergence to capacity. A polynomial dependence is implicit in [16], but obtaining a linear dependence requires lot more care. For the case  $q = 2$ , Theorem 2.1 is relatively easy to establish, as it is known that the extremal case (with minimal increase) occurs when  $H(X|Y = y) = H(X|Y)$  for all  $y$  in the support of  $Y$  [17, Lem 2.2]. This is based on the so-called ‘‘Mrs. Gerber’s Lemma’’ for binary-input channels [23, 22], the analog of which is not known for the non-binary case [10]. This allows us to reduce the binary version of (1) to an inequality about simple Bernoulli random variables with no conditioning, and the inequality then follows, as the sum of two  $p$ -biased coins is  $2p(1 - p)$ -biased and has higher entropy (unless  $p \in \{0, \frac{1}{2}, 1\}$ ). In the  $q$ -ary case, no such simple characterization of the extremal cases is known or seems likely [17, Sec 4.1]. Nevertheless, we prove the inequality in the  $q$ -ary setting by first proving two inequalities for unconditioned random variables, and then handling the conditioning explicitly based on several cases.

More specifically, the proof technique for Theorem 2.1 involves using an *averaging* argument to write the left-hand side of (1) as the expectation, over  $y, z \sim Y$ , of  $\Delta_{y,z} = H(X_y + X_z) - \frac{H(X_y) + H(X_z)}{2}$ , the entropy increase in the sum of random variables  $X_y$  and  $X_z$  with respect to their average entropy (this increase is called the *Ruzsa distance* between the random variables  $X_y$  and  $X_z$ , see [20]). We then rely on inequalities for *unconditioned* random variables to obtain a lower bound for this entropy increase. In general, one needs the entropy increase to be at least  $c \cdot \min\{H(X_y)(1 - H(X_y)), H(X_z)(1 - H(X_z))\}$ , but for some cases, we actually need such an entropy increase with respect to a larger *weighted* average. Hence, we prove the stronger inequality given by Theorem 4.10, which shows such an increase with respect to  $\frac{2H(X_y) + H(X_z)}{3}$  for  $H(X_y) \geq H(X_z)$ <sup>3</sup>. Moreover, for some cases

<sup>3</sup> While the weaker inequality  $H(A + B) \geq \frac{H(A) + H(B)}{2} + c \cdot \min\{H(A)(1 - H(A)), H(B)(1 - H(B))\}$  seems to be insufficient for our approach, it should be noted that the stronger inequality  $H(A + B) \geq \max\{H(A), H(B)\} + c \cdot \min\{H(A)(1 - H(A)), H(B)(1 - H(B))\}$  is generally not true. Thus, Theorem 4.10

of the proof, it suffices to bound  $\Delta_{y,z}$  from below by  $\frac{|H(X_y) - H(X_z)|}{2}$ , which is provided by Lemma 4.9, another inequality for unconditional random variables.

We note a version of Theorem 2.1 (in fact with tight bounds) for the case of unconditioned random variables  $X$  taking values in a torsion-free group was established by Tao in his work on entropic analogs of fundamental sumset inequalities in additive combinatorics [20] (results of similar flavor for integer-valued random variables were shown in [8]). Theorem 2.1 is a result in the same spirit for groups with torsion (and which further handles conditional entropy). While we do not focus on optimizing the dependence of  $\alpha(q)$  on  $q$ , pinning down the optimal dependence, especially for the case without any conditioning, seems like a natural question; see Remark 4.2 for further elaboration.

Given the entropy sumset inequality for conditional random variables, we are able to track the decay of  $\sqrt{H_n(1 - H_n)}$  and use Theorem 2.1 to show that for  $N = \text{poly}(1/\epsilon)$ , at most  $H(X) + \epsilon$  of the conditional entropies  $H(U_i|U_0, \dots, U_{i-1})$  exceed  $\epsilon$ . However, to construct a good source code, we need  $H(X) + \epsilon$  fraction of the conditional entropies to be  $\ll 1/N$ . This is achieved by augmenting a “fine” polarization stage that is analyzed using an appropriate Bhattacharyya parameter. The details of this step are similar to the binary case and appear in the full version of this paper [6].

The efficient construction of the linear source code (i.e., figuring out which entropies polarize very close to 0 so that those symbols can be dropped), and the efficient implementation of the successive cancellation decoder are similar to the binary case [7] and omitted here. Upon combining these ingredients, we get the following result on lossless compression with complexity scaling polynomially in the gap to capacity:

► **Theorem 2.2.** *Let  $X$  be a  $q$ -ary source for  $q$  prime with side information  $Y$  (which means  $(X, Y)$  is a correlated random variable). Let  $0 < \epsilon < \frac{1}{2}$ . Then there exists  $N \leq (1/\epsilon)^{c(q)}$  for a constant  $c(q) < \infty$  depending only on  $q$  and an explicit (constructible in  $\text{poly}(N)$  time) matrix  $L \in \{0, 1\}^{(H(X|Y) + \epsilon)N \times N}$  such that  $\vec{X} = (X_0, X_1, \dots, X_{N-1})^t$ , formed by taking  $N$  i.i.d. copies  $(X_0, Y_0), (X_1, Y_1), \dots, (X_{N-1}, Y_{N-1})$  of  $(X, Y)$ , can, with high probability, be recovered from  $L \cdot \vec{X}$  and  $\vec{Y} = (Y_0, Y_1, \dots, Y_{N-1})^t$  in  $\text{poly}(N)$  time.*

Moreover, can obtain Theorem 2.2 for *arbitrary* (not necessarily prime)  $q$  with the modification that the map  $\mathbb{Z}_q^N \rightarrow \mathbb{Z}_q^{H(X|Y) + \epsilon} N$  is no longer linear. This is obtained by factoring  $q$  into primes and combining polar codes over prime alphabets for each prime in the factorization.

**Channel coding.** Using known methods to construct channel codes from polar source codes for compressing sources with side information (see, for instance, [17, Sec 2.4] for a nice discussion of this aspect), we obtain the following result for channel coding, enabling reliable communication at rates within an additive gap  $\epsilon$  to the *symmetric capacity* for discrete memoryless channels over any fixed alphabet, with overall complexity bounded polynomially in  $1/\epsilon$ . Recall that a discrete memoryless channel (DMC)  $W$  has a finite input alphabet  $\mathcal{X}$  and a finite output alphabet  $\mathcal{Y}$  with transition probabilities  $p(y|x)$  for receiving  $y \in \mathcal{Y}$  when  $x \in \mathcal{X}$  is transmitted on the channel. The entropy  $H(W)$  of the channel is defined to be  $H(X|Y)$  where  $X$  is uniform in  $\mathcal{X}$  and  $Y$  is the output of  $W$  on input  $X$ ; the symmetric capacity of  $W$ , which is the largest rate at which one can reliably communicate on  $W$  when the inputs have a uniform prior, equals  $1 - H(W)$ . Moreover, it should be noted that if  $W$

---

provides the right middle ground. A limitation of similar spirit for the entropy increase when summing two integer-valued random variables was pointed out in [8].



is a *symmetric* DMC, then the symmetric capacity of  $W$  is precisely the Shannon capacity of  $W$ .

► **Theorem 2.3.** *Let  $q \geq 2$ , and let  $W$  be any discrete memoryless channel capacity with input alphabet  $\mathbb{Z}_q$ . Then, there exists an  $N \leq (1/\epsilon)^{c(q)}$  for a constant  $c(q) < \infty$  depending only on  $q$ , as well as a deterministic  $\text{poly}(N)$  construction of a  $q$ -ary code of block length  $N$  and rate at least  $1 - H(W) - \epsilon$ , along with a deterministic  $N \cdot \text{poly}(\log N)$  time decoding algorithm for the code such that the block error probability for communication over  $W$  is at most  $2^{-N^{0.49}}$ . Moreover, when  $q$  is prime, the constructed codes are linear.*

The structure of our paper will be as follows. Section 3 will introduce notation, describe the construction of polar codes, and define channels as a tool for analyzing entropy increases for a pair of correlated random variables. Section 4 will then present a sketch of our main theorem and describe the “rough” and “fine” polarization results that follow from the main theorem and allow us to achieve Theorem 2.2. Section 5 shows how polar codes for prime alphabets may be combined to obtain a capacity-achieving construction over all alphabets, thereby achieving a variant of Theorem 2.2 over non-prime alphabets, as well its channel-coding counterpart, Theorem 2.3.

### 3 Construction of Polar Codes

**Notation.** We begin by setting some of the notation to be used in the rest of the paper. We will let  $\lg$  denote the base 2 logarithm, while  $\ln$  will denote the natural logarithm.

For our purposes, unless otherwise stated,  $q$  will be a prime integer, and we identify  $\mathbb{Z}_q = \{0, 1, 2, \dots, q-1\}$  with the additive group of integers modulo  $q$ . We will generally view  $\mathbb{Z}_q$  as a  $q$ -ary alphabet.

Given a  $q$ -ary random variable  $X$  taking values in  $\mathbb{Z}_q$ , we let  $H(X)$  denote the *normalized entropy* of  $X$ :

$$H(X) = -\frac{1}{\lg q} \sum_{a \in \mathbb{Z}_q} \Pr[X = a] \lg(\Pr[X = a]).$$

In a slight abuse of notation, we also define  $H(p)$  for a probability distribution  $p$ . If  $p$  is a probability distribution over  $\mathbb{Z}_q$ , then we shall let  $H(p) = H(X)$ , where  $X$  is a random variable sampled according to  $p$ . Also, for nonnegative constants  $c_0, c_1, \dots, c_{q-1}$  summing to 1, we will often write  $H(c_0, \dots, c_{q-1})$  as the entropy of the probability distribution on  $\mathbb{Z}_q$  that samples  $i$  with probability  $c_i$ . Moreover, for a probability distribution  $p$  over  $\mathbb{Z}_q$ , we let  $p^{(+j)}$  denote the  $j^{\text{th}}$  *cyclic shift* of  $p$ , namely, the probability distribution  $p^{(+j)}$  over  $\mathbb{Z}_q$  that satisfies

$$p^{(+j)}(m) = p(m - j)$$

for all  $m \in \mathbb{Z}_q$ , where  $m - j$  is taken modulo  $q$ . Note that  $H(p) = H(p^{(+j)})$  for all  $j \in \mathbb{Z}_q$ .

Also, let  $\|\cdot\|_1$  denote the  $\ell_1$  norm on  $\mathbb{R}^q$ . In particular, for two probability distributions  $p$  and  $p'$ , the quantity  $\|p - p'\|_1$  will correspond to twice the total variational distance between  $p$  and  $p'$ .

Finally, given a row vector (tuple)  $\vec{v}$ , we let  $\vec{v}^t$  denote a column vector given by the transpose of  $\vec{v}$ .

### 3.1 Encoding Map

Let us formally define the polarization map that we will use to compress a source  $X$ . Given  $n \geq 1$ , we define an invertible linear transformation  $G : \mathbb{Z}_q^{2^n} \rightarrow \mathbb{Z}_q^{2^n}$  by  $G = G_n$ , where  $G_t : \mathbb{Z}_q^{2^t} \rightarrow \mathbb{Z}_q^{2^t}$ ,  $0 \leq t \leq n$  is a sequence of invertible linear transformations defined as follows:  $G_0$  is the identity map on  $\mathbb{Z}_q$ , and for any  $0 \leq k < n$  and  $\vec{X} = (X_0, X_1, \dots, X_{2^{k+1}-1})^t$ , we recursively define  $G_{k+1}\vec{X}$  as

$$G_{k+1}\vec{X} = \pi_{k+1}(G_k(X_0, \dots, X_{2^k-1}) + G_k(X_{2^k}, \dots, X_{2^{k+1}-1}), G_k(X_{2^k}, \dots, X_{2^{k+1}-1})),$$

where  $\pi_{k+1} : \mathbb{Z}_q^{2^{k+1}} \rightarrow \mathbb{Z}_q^{2^{k+1}}$  is a permutation such that  $\pi_n(v)_j = v_i$  for  $j = 2i$ , and  $\pi_n(v)_j = v_{i+2^k}$  for  $j = 2i + 1$ .

$G$  also has an explicit matrix form, namely,  $G = B_n K^{\otimes n}$ , where  $K = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$ ,  $\otimes$  is the Kronecker product, and  $B_n$  is the  $2^n \times 2^n$  bit-reversal permutation matrix for  $n$ -bit strings (see [3]).

In our set-up, we have a  $q$ -ary source  $X$ , and we let  $\vec{X} = (X_0, X_1, \dots, X_{2^n-1})^t$  be a collection of  $N = 2^n$  i.i.d. samples from  $X$ . Moreover, we encode  $\vec{X}$  as  $\vec{U} = (U_0, U_1, \dots, U_{2^n-1})^t$ , given by  $\vec{U} = G \cdot \vec{X}$ . Note that  $G$  only has 0, 1 entries, so each  $U_i$  is the sum (modulo  $q$ ) of some subset of the  $X_i$ 's.

### 3.2 Channels

For purposes of our analysis, we define a *channel*  $W = (A; B)$  to be a pair of correlated random variables  $A, B$ ; moreover, we define the *channel entropy* of  $W$  to be  $H(W) = H(A|B)$ , i.e., the entropy of  $A$  conditioned on  $B$ .<sup>4</sup>

Given a channel  $W$ , we can define two channel transformations – and + as follows. Suppose we take two i.i.d. copies  $(A_0; B_0)$  and  $(A_1; B_1)$  of  $W$ . Then,  $W^-$  and  $W^+$  are defined by

$$\begin{aligned} W^- &= (A_0 + A_1; B_0, B_1) \\ W^+ &= (A_1; A_0 + A_1, B_0, B_1). \end{aligned}$$

By the chain rule for entropy, we see that

$$H(W^-) + H(W^+) = 2H(W). \quad (2)$$

In other words, splitting two copies of  $W$  into  $W^-$  and  $W^+$  preserves the total channel entropy. These channels are easily seen to obey  $H(W^+) \leq H(W) \leq H(W^-)$ , and the key to our analysis will be quantifying the separation in the entropies of the two split channels.

The aforementioned channel transformations will help us abstract each step of the recursive polarization that occurs in the definition of  $G$ . Let  $W = (X; Y)$ , where  $X$  is a source taking values in  $\mathbb{Z}_q$ , and  $Y$  can be viewed as side information. Then,  $H(W) = H(X|Y)$ . One special case occurs when  $Y = 0$ , which corresponds to an absence of side information.

Note that if start with  $W$ , then after  $n$  successive applications of either  $W \mapsto W^-$  or  $W \mapsto W^+$ , we can obtain one of  $N = 2^n$  possible channels in  $\{W^s : s \in \{+, -\}^n\}$ . (Here, if

<sup>4</sup> It should be noted  $W$  can also be interpreted as a communication channel that takes in an input  $A$  and outputs  $B$  according to some conditional probability distribution. This is quite natural in the noisy channel coding setting in which one wishes to use a polar code for encoding data in order to achieve the channel capacity of a symmetric discrete memoryless channel. However, since we focus on the problem of source coding (data compression) rather than noisy channel coding in this paper, we will simply view  $W$  as a pair of correlated random variables.

$s = s_0 s_1 \cdots s_{n-1}$ , with each  $s_i \in \{+, -\}$ , then  $W^s$  denotes  $(\cdots ((W^{s_0})^{s_1}) \cdots)^{s_{n-2}})^{s_{n-1}}$ . By successive applications of (2), we know that

$$\sum_{s \in \{+, -\}^n} W^s = 2^n H(W) = 2^n H(X|Y).$$

Moreover, it can be verified (see [17]) that if  $0 \leq i < 2^n$  has binary representation  $\overline{b_{n-1} b_{n-2} \cdots b_0}$  (with  $b_0$  being the least significant bit of  $i$ ), then

$$H(U_i|U_0, U_1, \dots, U_{i-1}, Y_0, Y_1, \dots, Y_{N-1}) = H(W^{s_{n-1} s_{n-2} \cdots s_0}),$$

where  $s_j = -$  if  $b_j = 0$ , and  $s_j = +$  if  $b_j = 1$ . As shorthand notation, we will define the channel

$$W_n^{(i)} = W^{s_{n-1} s_{n-2} \cdots s_0},$$

where  $s_0, s_1, \dots, s_{n-1}$  are as above. [18] shows that all but a vanishing fraction of the  $N$  channels  $W^s$  will be have channel entropy close to 0 or 1:

► **Theorem 3.1.** *For any  $\delta > 0$ , we have that*

$$\lim_{n \rightarrow \infty} \frac{|\{s \in \{+, -\}^n : H(W^s) \in (\delta, 1 - \delta)\}|}{2^n} = 0.$$

Hence, one can then argue that as  $n$  grows, the fraction of channels with channel entropy close to 1 approaches  $H(X|Y)$ . In particular, for any  $\delta > 0$ , if we let

$$\text{High}_{n,\delta} = \{i : H(U_i|U_0, U_1, \dots, U_{i-1}, Y_0, Y_1, \dots, Y_{N-1}) > \delta\}, \quad (3)$$

then

$$\frac{|\text{High}_{n,\delta}|}{2^n} \rightarrow H(X|Y),$$

as  $n \rightarrow \infty$ . Thus, it can be shown that for any fixed  $\epsilon > 0$  and small  $\delta > 0$ , there exists suitably large  $n$  such that  $\{U_i\}_{i \in \text{High}_{n,\delta}}$  gives a source coding of  $\vec{X} = (X_0, X_1, \dots, X_{N-1})$  (with side information  $\vec{Y} = (Y_0, Y_1, \dots, Y_{N-1})$  with rate  $\leq H(X|Y) + \epsilon$ ).

Our goal is to show that  $N = 2^n$  can be taken to be just polynomial in  $1/\epsilon$  in order to obtain a rate  $\leq H(X|Y) + \epsilon$ .

### 3.3 Bhattacharyya Parameter

In order to analyze a channel  $W = (X; Y)$ , where  $X$  takes values in  $\mathbb{Z}_q$ , we will define the  $q$ -ary source Bhattacharyya parameter  $Z_{\max}(W)$  of the channel  $W$  as

$$Z_{\max}(W) = \max_{d \neq 0} Z_d(W),$$

where

$$Z_d(W) = \sum_{x \in \mathbb{Z}_q} \sum_{y \in \text{Supp}(Y)} \sqrt{p(x, y) p(x + d, y)}.$$

Here,  $p(x, y)$  is the probability that  $X = x$  and  $Y = y$  under the joint probability distribution  $(X, Y)$ .

Now, the *maximum likelihood decoder* attempts to decode  $x$  given  $y$  by choosing the most likely symbol  $\hat{x}$ :

$$\hat{x} = \arg \max_{x' \in \mathbb{Z}_q} \Pr[X = x' | Y = y].$$

Let  $P_e(W)$  be the probability of an error under maximum likelihood decoding, i.e., the probability that  $\hat{x} \neq x$  (or the defining  $\arg \max$  for  $\hat{x}$  is not unique) for random  $(x, y) \sim (X, Y)$ . It is known (see Proposition 4.7 in [17]) that  $Z_{\max}(W)$  provides an upper bound on  $P_e(W)$ :

► **Lemma 3.2.** *If  $W$  is a channel with  $q$ -ary input, then the error probability of the maximum-likelihood decoder for a single channel use satisfies  $P_e(W) \leq (q - 1)Z_{\max}(W)$ .*

Next, the following proposition shows how the  $Z_{\max}$  operator behaves on the polarized channels  $W^-$  and  $W^+$ . For a proof, see Proposition 4.16 in [17].

► **Lemma 3.3.**  $Z_{\max}(W^+) \leq Z_{\max}(W)^2$ , and  $Z_{\max}(W^-) \leq q^3 Z_{\max}(W)$ .

Finally, the following lemma shows that  $Z_{\max}(W)$  is small whenever  $H(W)$  is small.

► **Lemma 3.4.**  $Z_{\max}(W)^2 \leq (q - 1)^2 H(W)$ .

The proof follows from Proposition 4.8 of [17].

## 4 Quantification of Polarization

Our goal is to show “rough” polarization of the channel. More precisely, we wish to show that for some  $m = O(\lg(1/\epsilon))$  and constant  $K$ , we have  $\Pr_i[Z(W_m^{(i)}) \leq 2^{-Km}] \geq 1 - H(W) - \epsilon$ . The above polarization result will then be used to show the stronger notion of “fine” polarization, which will establish the polynomial gap to capacity.

The main ingredient in showing polarization is the following theorem, which quantifies the splitting that occurs with each polarizing step.

► **Theorem 4.1.** *For any channel  $W = (A; B)$ , where  $A$  takes values in  $\mathbb{Z}_q$ , we have*

$$H(W^-) \geq H(W) + \alpha(q) \cdot H(W)(1 - H(W)),$$

where  $\alpha(q)$  is a constant depending only on  $q$ .

Theorem 4.1 follows as a direct consequence of Theorem 2.1, whose proof we sketch in Section 4.2. Section 4.1 focuses on proving Theorem 4.10 (tackling the unconditioned case), which will be used in the proof of Theorem 2.1.

### 4.1 Unconditional Entropy Gain

In this section, we sketch some results (Lemma 4.9 and Theorem 4.10) that provide a lower bound on the normalized (unconditional) entropy  $H(A + B)$  of a sum of random variables  $A, B$  in terms of the individual entropies.

First, we present some lemmas, whose proofs appear in the full version of the paper [6] (with the exception of Lemma 4.3, which appears, with proof, as Lemma 4.5 in [17]).

Lemmas 4.2 and 4.3 are used to show that the entropy of a linear combination of cyclic shifts of a distribution exhibits an increase over the entropy of the original distribution.

► **Lemma 4.2.** *Let  $p$  be a distribution over  $\mathbb{Z}_q$ . Then, if  $\lambda_0, \lambda_1, \dots, \lambda_{q-1}$  are nonnegative numbers adding up to 1, we have*

$$H(\lambda_0 p^{(+0)} + \lambda_1 p^{(+1)} + \dots + \lambda_{q-1} p^{(+(q-1))}) \geq H(p) + \frac{1}{2 \lg q} \cdot \frac{\lambda_i \lambda_j}{\lambda_i + \lambda_j} \|p^{(+i)} - p^{(+j)}\|_1^2,$$

for any  $i \neq j$  such that  $\lambda_i + \lambda_j > 0$ .

► **Lemma 4.3.** *Let  $p$  be a distribution over  $\mathbb{Z}_q$ , where  $q$  is prime. Then,*

$$\|p^{(+i)} - p^{(+j)}\|_1 \geq \frac{(1 - H(p)) \lg q}{2q^2(q-1) \lg e}.$$

Lemmas 4.4 and 4.5 show matching lower and upper bounds (up to a constant factor) on the entropy of a low-entropy random variable with high mass on one symbol.

► **Lemma 4.4.** *Suppose  $0 < \epsilon < 1$ . If  $p$  is a distribution on  $\mathbb{Z}_q$  with mass  $1 - \epsilon$  on one symbol, then*

$$H(p) \geq \frac{\epsilon \lg(1/\epsilon)}{\lg q}.$$

► **Lemma 4.5.** *Suppose  $0 < \epsilon \leq \min\{\epsilon_1, \epsilon_2\}$ , where  $\epsilon_1 = \frac{1}{500}$  and  $\epsilon_2 = \frac{1}{(q-1)^4}$ . If  $p$  is a distribution on  $\mathbb{Z}_q$  with mass  $1 - \epsilon$  on one symbol, then*

$$H(p) \leq \frac{17\epsilon \lg(1/\epsilon)}{12 \lg q}.$$

Lemma 4.6 provides a lower bound on  $H(X + Y)$  for low-entropy random variables  $X, Y$  that each have most of their mass on a single symbol.

► **Lemma 4.6.** *Let  $X, Y$  be random variables taking values in  $\mathbb{Z}_q$  such that  $H(X) \geq H(Y)$ , and assume  $0 < \epsilon, \epsilon' \leq \min\{\epsilon_1, \epsilon_2\}$ , where  $\epsilon_1 = \frac{1}{500}$  and  $\epsilon_2 = \frac{1}{(q-1)^4}$ . Suppose that  $X$  has mass  $1 - \epsilon$  on one symbol, while  $Y$  has mass  $1 - \epsilon'$  on a symbol. Then,*

$$H(X + Y) - \frac{2H(X) + H(Y)}{3} \geq \frac{1}{51} \cdot H(Y)(1 - H(Y)). \quad (4)$$

Lemma 4.7 provides a lower bound as well as upper bound on high-entropy random variables that are close to uniform distributions.

► **Lemma 4.7.** *Suppose  $p$  is a distribution on  $\mathbb{Z}_q$  such that for each  $0 \leq i \leq q-1$ , we have  $p(i) = \frac{1}{q} + \delta_i$  with  $\max_{0 \leq i < q} |\delta_i| = \delta$ . Then,*

$$1 - \frac{q^2}{\ln q} \delta^2 \leq H(p) \leq 1 - \frac{q^2(q \ln q - (q-1))}{(q-1)^3 \ln q} \delta^2.$$

Lemma 4.8 shows provides a lower bound on  $H(X + Y)$  for high-entropy random variables  $X$  and  $Y$  that are both close to uniform distributions.

► **Lemma 4.8.** *Let  $X$  and  $Y$  be random variables taking values in  $\mathbb{Z}_q$  such that  $H(X) \geq H(Y)$ . Also, assume  $0 < \delta, \delta' \leq \frac{1}{2q^2}$ . Suppose  $\Pr[X = i] = \frac{1}{q} + \delta_i$  and  $\Pr[Y = i] = \frac{1}{q} + \delta'_i$  for  $0 \leq i \leq q-1$ , such that  $\max_{0 \leq i < q} |\delta_i| = \delta$  and  $\max_{0 \leq i < q} |\delta'_i| = \delta'$ . Then,*

$$H(X + Y) - H(X) \geq \frac{\ln q}{16q^2} \cdot H(X)(1 - H(X)). \quad (5)$$

Now, we present some results showing a lower bound on  $H(A + B)$  in terms of  $H(A)$  and  $H(B)$ .

► **Lemma 4.9.** *Let  $A$  and  $B$  be random variables taking values over  $\mathbb{Z}_q$ . Then,*

$$H(A + B) \geq \max\{H(A), H(B)\}.$$

**Proof.** Without loss of generality, assume  $H(A) \geq H(B)$ . Let  $p$  be the underlying probability distribution for  $A$ . Let  $\lambda_i = \Pr[B = i]$ . Then, the underlying probability distribution of  $A + B$  is  $\lambda_0 p^{(+0)} + \lambda_1 p^{(+1)} + \dots + \lambda_{q-1} p^{(+(q-1))}$ . The desired result then follows directly from Lemma 4.2. ◀

The next theorem provides a different lower bound for  $H(A + B)$ .

► **Theorem 4.10.** *Let  $A$  and  $B$  be random variables taking values over  $\mathbb{Z}_q$  such that  $H(A) \geq H(B)$ . Then,*

$$H(A + B) \geq \frac{2H(A) + H(B)}{3} + c \cdot \min\{H(A)(1 - H(A)), H(B)(1 - H(B))\}$$

for  $c = \frac{\gamma_0^3 \lg q}{48q^5(q-1)^3 \lg(6/\gamma_0) \lg^2 e}$ , where  $\gamma_0 = \frac{1}{500(q-1)^4 \lg q}$ .

The proof of this theorem appears in the full version of our paper [6], but we provide a brief overview of the proof below.

**Overview of proof.** The proof of Theorem 4.10 splits into various cases depending on where  $H(A)$  and  $H(B)$  lie. Note that some of these cases overlap. The overall idea is as follows. If  $H(A)$  and  $H(B)$  are both bounded away from 0 and 1 (Case 2), then the desired inequality follows from the concavity of the entropy function, using Lemmas 4.2 and 4.3 (note that this uses primality of  $q$ ). Another setting in which the inequality can be readily proven is when  $H(A) - H(B)$  is bounded away from 0 (which we deal with in Cases 4 and 5).

Thus, the remaining cases occur when  $H(A)$  and  $H(B)$  are either both small (Case 1) or both large (Case 3). In the former case, one can show that  $A$  must have most of its weight on a particular symbol, and similarly for  $B$  (note that this is why we must choose  $\gamma_0 \ll \frac{1}{\lg q}$ ; otherwise,  $A$  could be, for instance, supported uniformly on a set of size 2). Then, one can use the fact that a  $q$ -ary random variable having weight  $1 - \epsilon$  has entropy  $\Theta(\epsilon \log(1/\epsilon))$  (Lemmas 4.4 and 4.5) in order to prove the desired inequality (using Lemma 4.6).

For the latter case, we simply show that each of the  $q$  symbols of  $A$  must have weight close to  $1/q$ , and similarly for  $B$ . Then, we use the fact that such a random variable whose maximum deviation from  $1/q$  is  $\delta$  has entropy  $1 - \Theta(\delta^2)$  (Lemma 4.7) in order to prove the desired result (using Lemma 4.8). ◀

## 4.2 Conditional Entropy Gain

Theorem 4.1 now follows as a simple consequence of our main theorem, which we restate below. The proof appears in the full version of our paper [6], but we provide an overview of the proof below.

► **Theorem 2.1.** *Let  $(X_i, Y_i)$ ,  $i = 1, 2$  be i.i.d. copies of a correlated random variable  $(X, Y)$  with  $X$  supported on  $\mathbb{Z}_q$  for a prime  $q$ . Then for some  $\alpha(q) > 0$ ,*

$$H(X_1 + X_2 | Y_1, Y_2) - H(X | Y) \geq \alpha(q) \cdot H(X | Y)(1 - H(X | Y)). \quad (1)$$

► **Remark.** We have not attempted to optimize the dependence of  $\alpha(q)$  on  $q$ , and our proof gets  $\alpha(q) \geq \frac{1}{q^{O(1)}}$ . It is easy to see that  $\alpha(q) \leq O(1/\log q)$  even without conditioning (i.e., when  $Y = 0$ ). Understanding what is the true behavior of  $\alpha(q)$  seems like an interesting and basic question about sums of random variables. For random variables  $X$  taking values from a torsion-free group  $G$  and with sufficiently large  $H_2(X)$ , it is known that  $H_2(X_1 + X_2) - H_2(X) \geq \frac{1}{2} - o(1)$  and that this is best possible [20], where  $H_2(\cdot)$  denotes the *unnormalized* entropy (in bits). When  $G$  is the group of integers, a lower bound  $H_2(X_1 + X_2) - H_2(X) \geq g(H_2(X))$  for an increasing function  $g(\cdot)$  was shown for all  $\mathbb{Z}$ -valued random variables  $X$  [8]. For groups  $G$  with torsion, we cannot hope for any entropy increase unless  $G$  is finite and isomorphic to  $\mathbb{Z}_q$  for  $q$  prime (as  $G$  cannot have non-trivial finite subgroups), and we cannot hope for an absolute entropy increase even for  $\mathbb{Z}_q$ . So determining the asymptotics of  $\alpha(q)$  as a function of  $q$  is the analog of the question studied in [20] for finite groups.

**Overview of proof.** Let  $X_y$  denote  $X|Y = y$ . Then, we use an averaging argument: We reduce the desired inequality to providing a lower bound for  $\Delta_{y,z} = H(X_y + X_z) - \frac{H(X_y) + H(X_z)}{2}$ , whose expectation over  $y, z \sim Y$  is the left-hand side of (1). Then, one splits into three cases for small, large, and medium values of  $H(X|Y)$ .

Thus, we reduce the problem to arguing about unconditional entropies. As a first step, one would expect to prove  $\Delta_{y,z} \geq \min\{H(X_y)(1 - H(X_y)), H(X_z)(1 - H(X_z))\}$  and use this in the proof of the conditional inequality. However, this inequality turns out to be too weak to deal with the case in which  $H(X|Y)$  is tiny (case 2). This is the reason we require Theorem 4.10, which provides an increase for  $H(X_y + X_z)$  over a higher *weighted* average instead of the simple average of  $H(X_y)$  and  $H(X_z)$ . Additionally, we use the inequality  $H(X_y + X_z) \geq \max\{H(X_y), H(X_z)\}$  to handle certain cases, and this is provided by Lemma 4.9.

In cases 1 and 3 (for  $H(X|Y)$  in the middle and high regimes), the proof idea is that either (1) there is a significant mass of  $(y, z) \sim Y \times Y$  for which  $H(X_y)$  and  $H(X_z)$  are separated, in which case one can use Lemma 4.9 to bound  $\mathbf{E}[\Delta_{y,z}]$  from below, or (2) there is a significant mass of  $y \sim Y$  for which  $H(X_y)$  lies away from 0 and 1, in which case  $H(X_y)(1 - H(X_y))$  can be bounded from below, enabling us to use Theorem 4.10. ◀

### 4.3 Rough Polarization

Now that we have established Theorem 4.1, we are ready to show rough polarization of the channels  $W_n^{(i)}$ ,  $0 \leq i < 2^n$ , for large enough  $n$ . The precise theorem showing rough polarization is as follows.

► **Theorem 4.11.** *There is a constant  $\Lambda < 1$  such that the following holds. For any  $\Lambda < \rho < 1$ , there exists a constant  $b_\rho$  such that for all channels  $W$  with  $q$ -ary input, all  $\epsilon > 0$ , and all  $n > b_\rho \lg(1/\epsilon)$ , there exists a set*

$$\mathcal{W}' \subseteq \{W_n^{(i)} : 0 \leq i \leq 2^n - 1\}$$

*such that for all  $M \in \mathcal{W}'$ , we have  $Z_{\max}(M) \leq 2\rho^n$  and  $\Pr_i[W_n^{(i)} \in \mathcal{W}'] \geq 1 - H(W) - \epsilon$ .*

The proof of Theorem 4.11 follows from the following lemma:

► **Lemma 4.12.** *Let  $T(W) = H(W)(1 - H(W))$  denote the symmetric entropy of a channel  $W$ . Then, there exists a constant  $\Lambda < 1$  (possibly dependent on  $q$ ) such that*

$$\frac{1}{2} \left( \sqrt{T(W_{n+1}^{(2j)})} + \sqrt{T(W_{n+1}^{(2j+1)})} \right) \leq \Lambda \sqrt{T(W_n^{(j)})} \quad (6)$$

*for any  $0 \leq j < 2^n$ .*

The proofs of Theorem 4.11 and 4.12 follow from arguments similar to those found in the proofs of Proposition 5 and Lemma 8 in [7], except that we work with  $Z_{\max}$ . For completeness, the proofs appear in the full version of this paper [6].

#### 4.4 Fine Polarization

Now, we describe the statement of “fine polarization.” This is quantified by the following theorem.

► **Theorem 4.13.** *For any  $0 < \delta < \frac{1}{2}$ , there exists a constant  $c_\delta$  that satisfies the following statement: For any  $q$ -ary input memoryless channel  $W$  and  $0 < \epsilon < \frac{1}{2}$ , if  $n_0 > c_\delta \lg(1/\epsilon)$ , then*

$$\Pr_i \left[ Z_{\max}(W_{n_0}^{(i)}) \leq 2^{-2^{\delta n_0}} \right] \geq 1 - H(W) - \epsilon.$$

The proof follows from arguments similar to those in [4, 7] and appears in the full version of the paper [6].

As a corollary, we obtain the following result on lossless compression with complexity scaling polynomially in the gap to capacity:

► **Theorem 2.2.** *Let  $X$  be a  $q$ -ary source for  $q$  prime with side information  $Y$  (which means  $(X, Y)$  is a correlated random variable). Let  $0 < \epsilon < \frac{1}{2}$ . Then there exists  $N \leq (1/\epsilon)^{c(q)}$  for a constant  $c(q) < \infty$  depending only on  $q$  and an explicit (constructible in  $\text{poly}(N)$  time) matrix  $L \in \{0, 1\}^{(H(X|Y)+\epsilon)N \times N}$  such that  $\vec{X} = (X_0, X_1, \dots, X_{N-1})^t$ , formed by taking  $N$  i.i.d. copies  $(X_0, Y_0), (X_1, Y_1), \dots, (X_{N-1}, Y_{N-1})$  of  $(X, Y)$ , can, with high probability, be recovered from  $L \cdot \vec{X}$  and  $\vec{Y} = (Y_0, Y_1, \dots, Y_{N-1})^t$  in  $\text{poly}(N)$  time.*

**Proof.** Let  $W = (X; Y)$ , and fix  $\delta = 0.499$ . Also, let  $N = 2^{n_0}$ . Then, by Theorem 4.13, for any  $n_0 > c_\delta \lg(1/\epsilon)$ , we have that

$$\Pr_i \left[ Z_{\max}(W_{n_0}^{(i)}) \leq 2^{-2^{\delta n_0}} \right] \geq 1 - H(X) - \epsilon.$$

Moreover, let  $N = 2^{n_0}$ . Recall the notation in (3). Then, letting  $\delta' = 2^{-2^{\delta n_0}}$ , we have that  $\Pr_i[i \in \text{High}_{n_0, \delta'}] \leq H(X|Y) + \epsilon$  and  $Z(W_{n_0}^{(i)}) \geq \delta'$  for all  $i \in \text{High}_{n_0, \delta'}$ . Thus, we can take  $L$  to be the linear map  $G_{n_0}$  projected onto the coordinates of  $\text{High}_{n_0, \delta'}$ .

By Lemma 3.2 and the union bound, the probability that attempting to recover  $\vec{X}$  from  $L \cdot \vec{X}$  and  $\vec{Y}$  results in an error is given by

$$\sum_{i \notin \text{High}_{n_0, \delta'}} P_e(W_{n_0}^{(i)}) \leq \sum_{i \notin \text{High}_{n_0, \delta'}} (q-1) Z_{\max}(W_{n_0}^{(i)}) \leq (q-1) N \delta' = (q-1) 2^{n_0 - 2^{\delta n_0}}, \quad (7)$$

which is  $\leq 2^{-N^{0.49}}$  for  $N \geq (1/\epsilon)^\mu$  for some positive constant  $\mu$  (possibly depending on  $q$ ). Hence, it suffices to take  $c(q) = 1 + \max\{c_\delta, \mu\}$ .

Finally, the fact that both the construction of  $L$  and the recovery of  $\vec{X}$  from  $L \cdot \vec{X}$  and  $\vec{Y}$  can be done in  $\text{poly}(N)$  time follows in a similar fashion to the binary case (see the binning algorithm and the successive cancellation decoder in [7] for details). Also, the entries of  $L$  are all in  $\{0, 1\}$  since  $L$  can be obtained by taking a submatrix of  $B_n K^{\otimes n_0}$ , where  $B_n$  is a permutation matrix, and  $K = \begin{pmatrix} 1 & \\ & 1 \end{pmatrix}$  (see [3]). ◀



## 5 Extension to Arbitrary Alphabets

In the previous sections, we have shown polarization and polynomial gap to capacity for polar codes over *prime* alphabets. We now describe how to extend this to obtain channel polarization and the explicit construction of a polar code with polynomial gap to capacity over *arbitrary* alphabets.

The idea is to use the multi-level code construction technique sketched in [18] (and also recently in [14] for alphabets of size  $2^m$ ). We outline the procedure here. Suppose we have a channel  $W = (X; Y)$ , where  $X \in \mathbb{Z}_q$  and  $Y \in \mathcal{Y}$ . Moreover, assume that  $q = \prod_{i=1}^s q_i$  is the prime factorization of  $q$ .

Now, we can write  $X = (U^{(1)}, U^{(2)}, \dots, U^{(s)})$ , where each  $U^{(i)}$  is a random variable distributed over  $[q_i]$ . We also define the channels  $W^{(1)}, W^{(2)}, \dots, W^{(s)}$  by  $W^{(j)} = (U^{(j)}; Y, U^{(1)}, U^{(2)}, \dots, U^{(j-1)})$ . Note that

$$\begin{aligned} H(W) &= H(X|Y) = H(U^{(1)}, U^{(2)}, \dots, U^{(s)}|Y) \\ &= \sum_{j=1}^s H(U^{(j)}|Y, U^{(1)}, U^{(2)}, \dots, U^{(j-1)}) \\ &= \sum_{j=1}^s H(W^{(j)}), \end{aligned}$$

which means that  $W$  splits into  $W^{(1)}, W^{(2)}, \dots, W^{(s)}$ . Since each  $W^{(j)}$  is a channel whose input is over a prime alphabet, one can polarize each  $W^{(j)}$  separately using the procedure of the previous sections. More precisely, the encoding procedure is as follows. For  $N$  large enough (as specified by Theorem 2.2), we take  $N$  copies  $(X_0; Y_0), (X_1; Y_1), \dots, (X_{N-1}; Y_{N-1})$  of  $W$ , where  $X_i = (U_i^{(1)}, U_i^{(2)}, \dots, U_i^{(s)})$ . Then, sequentially for  $j = 1, 2, \dots, s$ , we encode  $U_0^{(j)}, U_1^{(j)}, \dots, U_{N-1}^{(j)}$  using  $\left\{ \left( Y_i, U_i^{(1)}, U_i^{(2)}, \dots, U_i^{(j-1)} \right) \right\}_{i=0,1,\dots,N-1}$  as side information (which can be done by the procedure in previous sections, since  $U_j$  is a source over a prime alphabet).

For decoding, one can simply use  $s$  stages of the successive cancellation decoder. In the  $j^{\text{th}}$  stage, one uses the successive cancellation decoder for  $W^{(j)}$  in order to decode  $U_0^{(j)}, U_1^{(j)}, \dots, U_{N-1}^{(j)}$ , assuming that  $\left\{ U_i^{(k)} \right\}_{k < j}$  has been recovered correctly from the previous stages of successive cancellation decoding. Note that the error probability in decoding  $X_0, X_1, \dots, X_{N-1}$  can be obtained by taking a union bound over the error probabilities for each of the  $s$  stages of successive cancellation decoding. Since each individual error probability is exponentially small (see (7)), it follows that the overall error probability is also negligible.

As a consequence, we obtain Theorem 2.2 for non-prime  $q$ , with the additional modification that the map  $\mathbb{Z}_q^N \rightarrow \mathbb{Z}_q^{H(X|Y)+\epsilon}N$  is not linear. Moreover, using the translation from source coding to noisy channel coding (see [17, Sec 2.4]), we obtain the following result for channel coding.

► **Theorem 2.3.** *Let  $q \geq 2$ , and let  $W$  be any discrete memoryless channel capacity with input alphabet  $\mathbb{Z}_q$ . Then, there exists an  $N \leq (1/\epsilon)^{c(q)}$  for a constant  $c(q) < \infty$  depending only on  $q$ , as well as a deterministic  $\text{poly}(N)$  construction of a  $q$ -ary code of block length  $N$  and rate at least  $1 - H(W) - \epsilon$ , along with a deterministic  $N \cdot \text{poly}(\log N)$  time decoding algorithm for the code such that the block error probability for communication over  $W$  is at most  $2^{-N^{0.49}}$ . Moreover, when  $q$  is prime, the constructed codes are linear.*

► **Remark.** If  $q$  is prime, then the  $q$ -ary code of Theorem 2.3 is, in fact, linear.

**Acknowledgments.** We thank Emmanuel Abbe, Eren Şaşoğlu, and Patrick Xia for useful discussions about various aspects of polar codes.

---

### References

- 1 Emmanuel Abbe and Emre Telatar. Polar codes for the  $m$ -user multiple access channel. *IEEE Transactions on Information Theory*, 58(8):5437–5448, 2012.
- 2 Erdal Arıkan. Channel polarization: a method for constructing capacity-achieving codes for symmetric binary-input memoryless channels. *IEEE Transactions on Information Theory*, 55(7):3051–3073, 2009.
- 3 Erdal Arıkan. Source polarization. In *Proceedings of 2010 IEEE International Symposium on Information Theory*, pages 899–903, 2010.
- 4 Erdal Arıkan and Emre Telatar. On the rate of channel polarization. In *Proceedings of 2009 IEEE International Symposium on Information Theory*, pages 1493–1495, 2009.
- 5 Naveen Goela, Emmanuel Abbe, and Michael Gastpar. Polar codes for broadcast channels. In *Proceedings of the 2013 IEEE International Symposium on Information Theory, Istanbul, Turkey, July 7-12, 2013*, pages 1127–1131, 2013.
- 6 Venkatesan Guruswami and Ameya Velingker. An entropy sumset inequality and polynomially fast convergence to shannon capacity over all alphabets. *Electronic Colloquium on Computational Complexity (ECCC)*, 21:165, 2014.
- 7 Venkatesan Guruswami and Patrick Xia. Polar codes: Speed of polarization and polynomial gap to capacity. In *FOCS*, pages 310–319, 2013. Full version to appear in *IEEE Trans. on Info. Theory*, Jan. 2015.
- 8 Saeid Haghighatshoar, Emmanuel Abbe, and İ. Emre Telatar. A new entropy power inequality for integer-valued random variables. *IEEE Transactions on Information Theory*, 60(7):3787–3796, 2014.
- 9 Seyed Hamed Hassani, Kasra Alishahi, and Rüdiger L. Urbanke. Finite-length scaling of polar codes. *CoRR*, abs/1304.4778, 2013.
- 10 Varun Jog and Venkat Anantharam. The entropy power inequality and mrs. gerber’s lemma for groups of order  $2^n$ . *IEEE Transactions on Information Theory*, 60(7):3773–3786, 2014.
- 11 Satish Babu Korada. Polar codes for Slepian-Wolf, Wyner-Ziv, and Gelfand-Pinsker. In *Proceedings of the 2010 IEEE Information Theory Workshop*, pages 1–5, 2010.
- 12 Satish Babu Korada, Eren Sasoglu, and Rüdiger L. Urbanke. Polar codes: Characterization of exponent, bounds, and constructions. *IEEE Transactions on Information Theory*, 56(12):6253–6264, 2010.
- 13 Satish Babu Korada and Rüdiger L. Urbanke. Polar codes are optimal for lossy source coding. *IEEE Transactions on Information Theory*, 56(4):1751–1768, 2010.
- 14 Jingbo Liu and Emmanuel Abbe. Polynomial complexity of polar codes for non-binary alphabets, key agreement and slepian-wolf coding. In *48th Annual Conference on Information Sciences and Systems, CISS 2014, Princeton, NJ, USA, March 19-21, 2014*, pages 1–6, 2014. Available online at <http://arxiv.org/abs/1405.0776>.
- 15 Hessam Mahdavifar and Alexander Vardy. Achieving the secrecy capacity of wiretap channels using polar codes. *IEEE Transactions on Information Theory*, 57(10):6428–6443, 2011.
- 16 Eren Sasoglu. An entropy inequality for  $q$ -ary random variables and its application to channel polarization. In *ISIT*, pages 1360–1363. IEEE, 2010.
- 17 Eren Sasoglu. Polarization and polar codes. *Foundations and Trends in Communications and Information Theory*, 8(4):259–381, 2012.
- 18 Eren Sasoglu, Emre Telatar, and Erdal Arıkan. Polarization for arbitrary discrete memoryless channels. *CoRR*, abs/0908.0302, 2009.
- 19 Eren Sasoglu, Emre Telatar, and Edmund M. Yeh. Polar codes for the two-user multiple-access channel. *IEEE Transactions on Information Theory*, 59(10):6583–6592, 2013.

- 20 Terence Tao. Sunset and inverse sunset theory for Shannon entropy. *Combinatorics, Probability and Computing*, 19(4):603–639, 2010.
- 21 Lele Wang and Eren Sasoglu. Polar coding for interference networks. *CoRR*, abs/1401.7293, 2014.
- 22 Hans S. Witsenhausen. Entropy inequalities for discrete channels. *IEEE Transactions on Information Theory*, 20(5):610–616, 1974.
- 23 Aaron D. Wyner and Jacob Ziv. A theorem on the entropy of certain binary sequences and applications-I. *IEEE Transactions on Information Theory*, 19(6):769–772, 1973.

# The List-Decoding Size of Fourier-Sparse Boolean Functions

Ishay Haviv<sup>1</sup> and Oded Regev<sup>\*2</sup>

- 1 School of Computer Science  
The Academic College of Tel Aviv-Yaffo, Israel
- 2 Courant Institute of Mathematical Sciences  
New York University, USA

---

## Abstract

A function defined on the Boolean hypercube is *k-Fourier-sparse* if it has at most  $k$  nonzero Fourier coefficients. For a function  $f : \mathbb{F}_2^n \rightarrow \mathbb{R}$  and parameters  $k$  and  $d$ , we prove a strong upper bound on the number of  $k$ -Fourier-sparse Boolean functions that disagree with  $f$  on at most  $d$  inputs. Our bound implies that the number of uniform and independent random samples needed for learning the class of  $k$ -Fourier-sparse Boolean functions on  $n$  variables exactly is at most  $O(n \cdot k \log k)$ .

As an application, we prove an upper bound on the query complexity of testing Booleanity of Fourier-sparse functions. Our bound is tight up to a logarithmic factor and quadratically improves on a result due to Gur and Tamuz (Chicago J. Theor. Comput. Sci., 2013).

**1998 ACM Subject Classification** F.2.2 Nonnumerical Algorithms and Problems

**Keywords and phrases** Fourier-sparse functions, list-decoding, learning theory, property testing

**Digital Object Identifier** 10.4230/LIPIcs.CCC.2015.58

## 1 Introduction

Functions defined on the Boolean hypercube  $\{0, 1\}^n = \mathbb{F}_2^n$  are fundamental objects in theoretical computer science. It is well known that every such function  $f : \mathbb{F}_2^n \rightarrow \mathbb{R}$  can be represented as a linear combination

$$f = \sum_{S \subseteq [n]} \hat{f}(S) \cdot \chi_S$$

of the  $2^n$  functions  $\{\chi_S\}_{S \subseteq [n]}$  defined by  $\chi_S(x) = (-1)^{\sum_{i \in S} x_i}$ . This representation is known as the *Fourier expansion* of the function  $f$ , and the numbers  $\hat{f}(S)$  are known as its *Fourier coefficients*. The Fourier expansion of functions plays a central role in analysis of Boolean functions and finds applications in numerous areas of theoretical computer science including learning theory, property testing, hardness of approximation, social choice theory, and cryptography. For an in-depth introduction to the topic the reader is referred to the book of O’Donnell [22].

A classical result in learning theory is a general algorithm due to Kushilevitz and Mansour [19], based on results of Linial, Mansour, and Nisan [20] and Goldreich and

---

\* Supported by the Simons Collaboration on Algorithms and Geometry and by the National Science Foundation (NSF) under Grant No. CCF-1320188. 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 NSF.



Levin [12], which enables to efficiently learn classes of Boolean functions with a “simple” Fourier expansion. A common notion of simplicity of Fourier expansion is its sparsity. A function is said to be  $k$ -Fourier-sparse if it has at most  $k$  nonzero Fourier coefficients. It follows from [19] that given *query* access to a  $k$ -Fourier-sparse Boolean function  $f : \mathbb{F}_2^n \rightarrow \{0, 1\}$  it is possible to estimate its Fourier coefficients and to get a good approximation of  $f$  in running time polynomial in  $n$  and  $k$ . Later, it was shown that such running time even allows to reconstruct the function  $f$  exactly [13].

In recent years, properties of the Fourier expansion of functions were studied in the property testing framework. We now mention some of those results; since this will not be needed for the sequel, the reader can skip directly to the description of our results in the next section. Gopalan, O’Donnell, Servedio, Shpilka, and Wimmer considered in [13] the problem of testing if a given Boolean function is  $k$ -Fourier-sparse or  $\varepsilon$ -far from any such function. Another problem studied there is that of deciding if a function is  $k$ -Fourier-dimensional, that is, the Fourier support, viewed as a subset of  $\mathbb{F}_2^n$ , spans a subspace of dimension at most  $k$ , or  $\varepsilon$ -far from satisfying this property. Gopalan et al. [13] established testers for these properties whose query complexities depend only on  $k$  and  $\varepsilon$ . For  $k$ -Fourier-sparsity the query complexity was a certain polynomial in  $k$  and  $1/\varepsilon$  and for  $k$ -Fourier-dimensionality it was  $O(k \cdot 2^{2k}/\varepsilon)$ . They also proved lower bounds of  $\Omega(\sqrt{k})$  and  $\Omega(2^{k/2})$  respectively. Another parameter associated with Boolean functions is the degree of its representation as a polynomial over  $\mathbb{F}_2$ . The algorithmic task of testing if a function has  $\mathbb{F}_2$ -degree at most  $d$  or is  $\varepsilon$ -far from any such function was considered by Alon et al. [1] and then by Bhattacharyya et al. [6], who proved tight upper and lower bounds of  $\Theta(2^d + 1/\varepsilon)$  on the query complexity. Note that all the above properties fall into the class of linear-invariant properties, i.e., properties that are closed under compositions with any invertible linear transformation of the domain. These properties have recently attracted a significant amount of attention in the attempt to characterize efficient testability of them (see [24, 5] for related surveys).

## 1.1 Our Results

### List-decoding size

Our main technical result from which we derive all other results is concerned with the list-decoding size of Fourier-sparse Boolean functions. In general, the list-decoding problem of an error correcting code for a distance parameter  $d$  asks to find all the codewords whose Hamming distance from a given word is at most  $d$ . Here we consider the (non-linear) binary code of block length  $2^n$  whose codewords represent all the  $k$ -Fourier-sparse Boolean functions on  $n$  variables.

It is not difficult to show that the total number of such functions is at most  $2^{O(nk)}$ . Indeed, there are  $2^{O(nk)}$  ways to choose the support of  $\hat{f}$ , and  $2^{O(nk)}$  ways to set those Fourier coefficients which must all be integer multiples of  $2^{-n}$  in  $[-1, +1]$ . It is also not difficult to show that the distance between any two distinct codewords is at least  $2^n/k$ . Indeed, it is known that every  $k$ -Fourier-sparse Boolean function has  $\mathbb{F}_2$ -degree  $d \leq \log_2 k$  (see, e.g., [4, Lemma 3]), and therefore, by the Schwartz-Zippel lemma, every two distinct  $k$ -Fourier-sparse Boolean functions disagree on at least  $1/k$  fraction of the inputs. As a result, for every function  $f : \mathbb{F}_2^n \rightarrow \mathbb{R}$  there is at most one codeword of distance smaller than  $2^n/(2k)$  from  $f$ .

We are not aware of any other known bounds beyond those two naive ones. We address this question in the following theorem.

► **Theorem 1.1.** *For every function  $f : \mathbb{F}_2^n \rightarrow \mathbb{R}$ , the number of  $k$ -Fourier-sparse Boolean functions of distance at most  $d$  from  $f$  is  $2^{O(ndk \log k/2^n)}$ .*

We observe that for certain choices of  $k$  and  $d$  the bound given in Theorem 1.1 is tight. For example, let  $f$  be the constant zero function, let  $k < 2^{0.9n}$  be a power of 2, and take  $d = 2^n/k$ . Consider all the indicator functions of linear subspaces of  $\mathbb{F}_2^n$  of co-dimension  $\log_2 k$ . Every such function is of distance  $d$  from  $f$  and is  $k$ -Fourier-sparse (see Claim 2.4). The number of such functions is  $2^{\Theta(n \log k)} = 2^{\Theta(ndk \log k/2^n)}$ .

### Learning from samples

As an application of the list-decoding bound, we next consider the problem of learning the class of  $k$ -Fourier-sparse Boolean functions on  $n$  variables (exactly) from uniform and independent random *samples* (see, e.g., [2, 18] for related work). Let us note already at the outset that all the results mentioned here are *not* efficient: it is not known if there is an algorithm for the problem whose running time is some fixed polynomial in  $n$  times an arbitrary function of  $k$ . Among other things, such an algorithm would imply a breakthrough on the long-standing open question of learning juntas from samples [7, 21, 25, 18].

The question of recovering a function that is sparse in the Fourier (or other) basis from a few samples is the central question in the area of sparse recovery. It has been intensely investigated for over a decade and, among other things, has applications for compressed sensing and for the data stream model. The best previously known bounds on our question are  $O(n \cdot k \log^3 k) \leq O(n^4 \cdot k)$  due to Cheraghchi, Guruswami, and Velingker [11] and  $O(n^2 \cdot k \log k) \leq O(n^3 \cdot k)$  due to Bourgain [8], improving on a previous bound of Rudelson and Vershynin [23] (who themselves improved on the work of Candès and Tao [10]). We note in passing that they actually answer a harder question: first, because they handle all functions, not necessarily Boolean-valued; second, because they show that a randomly chosen set of sample locations of the above cardinality is good with high probability simultaneously *for all*  $k$ -Fourier-sparse functions (sometimes known as the “deterministic” setting), whereas we only want a random set of sample locations to be good with high probability for any *fixed*  $k$ -Fourier-sparse function (the “randomized” setting); finally, because they obtain the recovery result by proving a “restricted isometry property” of the Fourier matrix which among other things implies a recovery algorithm running in time polynomial in  $2^n$  and  $k$ .

Using Theorem 1.1, we improve the upper bound on the sample complexity of learning Fourier-sparse Boolean functions.

► **Corollary 1.2.** *The number of uniform and independent random samples required for learning the class of  $k$ -Fourier-sparse Boolean functions on  $n$  variables is  $O(n \cdot k \log k)$ .*

We believe that our better bound and its elementary proof shed more light on the problem and might be useful elsewhere. In fact, in a follow-up work [15] we employ the techniques developed here to study the “restricted isometry property” of random submatrices of Fourier (and other) matrices, improving on the aforementioned works [11, 8]. We finally note that a lower bound of  $\Omega(k \cdot (n - \log_2 k))$  on the sample complexity can be obtained by considering the problem of learning indicator functions of affine subspaces of  $\mathbb{F}_2^n$  of co-dimension  $\log_2 k$  (see Theorem 3.7; see, e.g., [3] for the same lower bound in a different setting).

### Testing Booleanity

We next consider the problem of testing Booleanity of Fourier-sparse functions, which was introduced and studied by Gur and Tamuz in [14]. In this problem, given access to a

$k$ -Fourier-sparse function  $f : \mathbb{F}_2^n \rightarrow \mathbb{R}$ , one has to decide if  $f$  is Boolean, i.e., its image is contained in  $\{0, 1\}$ , or not. The objective is to distinguish between the two cases with some constant probability using as few queries to  $f$  as possible. It was shown in [14] that there exists a (non-adaptive one-sided error) tester for the problem with query complexity  $O(k^2)$ , and that every tester for the problem has query complexity  $\Omega(k)$ . Here, we use our result on learning  $k$ -Fourier-sparse Boolean functions to improve the upper bound of [14] and prove the following.

► **Theorem 1.3.** *For every  $k$  there exists a non-adaptive one-sided error tester that using  $O(k \cdot \log^2 k)$  queries to an input  $k$ -Fourier-sparse function  $f : \mathbb{F}_2^n \rightarrow \mathbb{R}$  decides if  $f$  is Boolean or not with constant success probability.*

We note that, while the tester established in Theorem 1.3 has an improved query complexity, it is not clear if it is efficient with respect to running time. It can be shown, though, that using the learning algorithm of Fourier-sparse functions that follows from [8, 15] (instead of Corollary 1.2) in our proof of Theorem 1.3, one can obtain an efficient algorithm (running in time polynomial in  $n$  and  $k$ ) with the slightly worse query complexity of  $O(k \cdot \log^3 k)$ .

Finally, we complement Theorem 1.3 by the following nearly matching lower bound.

► **Theorem 1.4.** *Every non-adaptive one-sided error tester for Booleanity of  $k$ -Fourier-sparse functions has query complexity  $\Omega(k \cdot \log k)$ .*

## 1.2 Overview of Proofs

### 1.2.1 The List-Decoding Size of Fourier-Sparse Boolean Functions

In order to prove Theorem 1.1, we have to bound from above the number of  $k$ -Fourier-sparse Boolean functions of distance at most  $d$  from a general function  $f : \mathbb{F}_2^n \rightarrow \mathbb{R}$ . In the discussion below, let us consider the special case where  $f$  is the constant zero function. The general result follows easily.

Here, we have to bound the number of  $k$ -Fourier-sparse Boolean functions  $g : \mathbb{F}_2^n \rightarrow \{0, 1\}$  of support size at most  $d$ . We start by observing using Parseval's theorem that such functions have small spectral norm  $\|\hat{g}\|_1 = \sum_{S \subseteq [n]} |\hat{g}(S)|$ . Next, we observe that the Fourier expansion of the normalized function  $g/\|g\|_1$  is a convex combination of functions  $\pm \chi_S$ , and thus can be viewed, following a technique of Bruck and Smolensky [9], as an expectation over a distribution on the  $S$ 's. Using the Chernoff-Hoeffding bound and the bound on the spectral norm, we obtain a succinct representation for every such function  $g$ . The ability to represent these functions by a binary string of bounded length yields the upper bound on their number. We note that the proof approach somewhat resembles that of the upper bound on the list-decoding size of Reed-Muller codes due to Kaufman, Lovett, and Porat [17].

### 1.2.2 Learning Fourier-Sparse Boolean Functions

As a warmup, let us mention an easy upper bound of  $O(n \cdot k^2)$ . This follows by recalling that there are at most  $2^{O(nk)}$   $k$ -Fourier-sparse Boolean functions, and that each one differs from any fixed function on at least  $1/k$  fraction of the inputs. Hence by the union bound, after  $O(n \cdot k^2)$  samples all other functions will be eliminated.

The improved bound in Corollary 1.2 follows similarly using the list-decoding result of Theorem 1.1. Namely, we apply the union bound separately on functions of different distances from the input function. Functions that are nearby are harder to “hit” using random samples, but by the theorem, there are few of them; functions that are further away are in abundance, but they are easier to “hit” using random samples.

### 1.2.3 Testing Booleanity of Fourier-Sparse Functions

The testing Booleanity problem is somewhat different from typical property testing problems. Indeed, in property testing one usually has to distinguish objects that satisfy a certain property from those that are  $\varepsilon$ -far from the property for some distance parameter  $\varepsilon > 0$ . However, here the tester is required to decide if the function satisfies the Booleanity property or not, with no distance parameter involved. This unusual setting makes sense in this case because Fourier-sparse non-Boolean functions are always quite far from every Boolean function. More precisely, the authors of [14] used the uncertainty principle (see Proposition 2.1) to prove that every  $k$ -Fourier-sparse non-Boolean function  $f : \mathbb{F}_2^n \rightarrow \mathbb{R}$  is non-Boolean on at least  $\Omega(2^n/k^2)$  inputs (see Claim 2.3). This immediately implies a (non-adaptive one-sided error) tester that uses  $O(k^2)$  queries: just check that  $f$  is Boolean on  $O(k^2)$  uniform inputs in  $\mathbb{F}_2^n$ .

The analysis of [14] turns out to be tight, as there are  $k$ -Fourier-sparse non-Boolean functions that are not Boolean at only  $\Theta(2^n/k^2)$  points. Indeed, for an even integer  $n$ , consider the function  $f : \mathbb{F}_2^n \rightarrow \{0, 1, 2\}$  defined by

$$f(x_1, \dots, x_n) = \text{AND}(x_1, \dots, x_{n/2}) + \text{AND}(x_{n/2+1}, \dots, x_n), \quad (1)$$

which is not Boolean at only one point and has Fourier-sparsity  $2 \cdot 2^{n/2}$  (see Claim 2.4).

#### Upper bound

We prove Theorem 1.3 using our learning result, Corollary 1.2. To do so, we first observe that a restriction of a  $k$ -Fourier-sparse non-Boolean function to a random subspace of dimension  $O(\log k)$  is non-Boolean with high probability (see Lemma 4.1). Since a restriction to a subspace does not increase the Fourier-sparsity, this reduces our problem to testing Booleanity of  $k$ -Fourier-sparse functions on  $n = O(\log k)$  variables. Then, after  $O(k \cdot \log^2 k)$  samples from the subspace, if a non-Boolean value was found then we are clearly done. Otherwise, by Corollary 1.2, the samples uniquely specify a Boolean candidate for the restricted function. Such a function must be quite far from every other  $k$ -Fourier-sparse function (Boolean or not; see Claim 2.2). This enables us to decide if the restricted function equals the Boolean candidate function or not.

#### Lower bound

The upper bound in Theorem 1.3 gets close to the  $\Omega(k)$  lower bound proven by Gur and Tamuz in [14]. For their lower bound, they considered the following two distributions: (a) the uniform distribution over all Boolean  $n$ -variable functions that depend only on their first  $\log_2 k$  variables; (b) the uniform distribution over all  $n$ -variable functions that depend only on their first  $\log_2 k$  variables and return a Boolean value on  $k - 1$  of the assignments to the relevant variables and the value 2 otherwise. It can be easily seen that any (possibly adaptive) tester that distinguishes with some constant probability between distributions (a) and (b) has query complexity  $\Omega(k)$ . Since the first distribution is supported on  $k$ -Fourier-sparse Boolean functions and the second on  $k$ -Fourier-sparse non-Boolean functions, this implies that the same lower bound holds for the query complexity of testing Booleanity of  $k$ -Fourier-sparse functions.

Note that the distributions considered above are supported on  $\log_2 k$ -Fourier-dimensional functions. It can be seen (say, using the uncertainty principle) that such functions are not Boolean on at least  $1/k$  fraction of their inputs, so  $O(k)$  random samples suffice for finding a non-Boolean value if exists. Hence, in order to get beyond the  $\Omega(k)$  lower bound, we need



to consider  $k$ -Fourier-sparse functions that are not Boolean at only  $o(1/k)$  fraction of the inputs – our functions will actually have  $O(1/k^2)$  fraction of such inputs.

Specifically, we consider the distribution of functions obtained by composing the function  $f$  given in (1) with a random invertible affine transformation. This is the class of functions that can be represented as a sum  $\mathbf{1}_{V_1} + \mathbf{1}_{V_2}$  of two indicators of affine subspaces  $V_1, V_2 \subseteq \mathbb{F}_2^n$  of dimension  $n/2$ , which intersect at exactly one point. Intuitively, it seems that distinguishing the functions in this class from those where  $V_1$  and  $V_2$  have empty intersection requires the tester to learn the affine subspaces  $V_1$  and  $V_2$ , a task that requires  $\Omega(n \cdot 2^{n/2})$  queries. We prove such a lower bound for non-adaptive one-sided error testers. Since the above functions are  $k$ -Fourier-sparse for  $k = O(2^{n/2})$ , the obtained lower bound is  $\Omega(k \cdot \log k)$ .

## 2 Preliminaries

Let  $[n]$  denote the set  $\{1, \dots, n\}$ . A function  $f : \mathbb{F}_2^n \rightarrow \mathbb{R}$  is *Boolean* if its image is contained in  $\{0, 1\}$  and is *non-Boolean* otherwise. The *distance* between two functions  $f, g : \mathbb{F}_2^n \rightarrow \mathbb{R}$ , denoted  $\text{dist}(f, g)$ , is the number of vectors  $x \in \mathbb{F}_2^n$  for which  $f(x) \neq g(x)$ .

### Fourier Expansion

For every  $S \subseteq [n]$ , let  $\chi_S : \mathbb{F}_2^n \rightarrow \{-1, 1\}$  denote the function defined by  $\chi_S(x) = (-1)^{\sum_{i \in S} x_i}$ . It is well known that the  $2^n$  functions  $\{\chi_S\}_{S \subseteq [n]}$  form an orthonormal basis of the space of functions  $\mathbb{F}_2^n \rightarrow \mathbb{R}$  with respect to the inner product  $\langle f, g \rangle = \mathbb{E}_x[f(x) \cdot g(x)]$ , where  $x$  is distributed uniformly over  $\mathbb{F}_2^n$ . Thus, every function  $f : \mathbb{F}_2^n \rightarrow \mathbb{R}$  can be uniquely represented as a linear combination  $f = \sum_{S \subseteq [n]} \hat{f}(S) \cdot \chi_S$  of this basis. This representation is called the *Fourier expansion* of  $f$ , and the numbers  $\hat{f}(S)$  are referred to as its *Fourier coefficients*. The support of  $f$  is defined by  $\text{supp}(f) = \{x \in \mathbb{F}_2^n \mid f(x) \neq 0\}$  and the support of  $\hat{f}$ , known as the *Fourier spectrum* of  $f$ , by  $\text{supp}(\hat{f}) = \{S \subseteq [n] \mid \hat{f}(S) \neq 0\}$ . We say that  $f$  is  $k$ -Fourier-sparse<sup>1</sup> if  $|\text{supp}(\hat{f})| \leq k$ . For every  $p \geq 1$  we denote  $\|\hat{f}\|_p = (\sum_{S \subseteq [n]} |\hat{f}(S)|^p)^{1/p}$ . For  $p = 1$ ,  $\|\hat{f}\|_1$  is known as the *spectral norm* of  $f$ . *Parseval's theorem* states that  $\mathbb{E}_x[f(x)^2] = \|\hat{f}\|_2^2$ .

The uncertainty principle says that there is no nonzero function  $f$  for which the supports of both  $f$  and  $\hat{f}$  are small (see, e.g., [22, Exercise 3.15]). We state it below with two simple consequences.

► **Proposition 2.1** (The Uncertainty Principle). *For every nonzero function  $f : \mathbb{F}_2^n \rightarrow \mathbb{R}$ ,*

$$|\text{supp}(f)| \cdot |\text{supp}(\hat{f})| \geq 2^n.$$

► **Claim 2.2.** *For every two distinct  $k$ -Fourier-sparse functions  $f, g : \mathbb{F}_2^n \rightarrow \mathbb{R}$ ,  $\text{dist}(f, g) \geq 2^n/(2k)$ .*

**Proof.** Apply Proposition 2.1 to the function  $f - g$ , whose Fourier-sparsity is at most  $2k$ . ◀

► **Claim 2.3** ([14]). *For every  $k$ -Fourier-sparse function  $f : \mathbb{F}_2^n \rightarrow \mathbb{R}$ , if  $f$  is non-Boolean then*

$$|\{x \in \mathbb{F}_2^n \mid f(x) \notin \{0, 1\}\}| \geq \frac{2}{k^2 + k + 2} \cdot 2^n.$$

<sup>1</sup> Boolean functions are sometimes defined in the literature with range  $\{-1, +1\}$  rather than  $\{0, 1\}$ . Notice that this affects the Fourier-sparsity by at most 1.

**Proof.** Apply Proposition 2.1 to the function  $f \cdot (f - 1)$ , whose Fourier-sparsity is at most

$$|\{S \Delta T \mid S, T \in \text{supp}(\hat{f})\}| + |\text{supp}(\hat{f})| \leq \binom{k}{2} + k + 1,$$

where  $\Delta$  stands for symmetric difference of sets. ◀

We also need the following simple claim.

► **Claim 2.4.** *For every affine subspace  $V \subseteq \mathbb{F}_2^n$  of co-dimension  $k$ , the indicator function  $\mathbf{1}_V : \mathbb{F}_2^n \rightarrow \{0, 1\}$  is  $2^k$ -Fourier-sparse.*

**Proof.** Since  $V$  has co-dimension  $k$ , there exist  $a_1, \dots, a_k \in \mathbb{F}_2^n$  and  $b_1, \dots, b_k \in \mathbb{F}_2$  such that  $V = \{x \in \mathbb{F}_2^n \mid \langle x, a_i \rangle = b_i, i = 1, \dots, k\}$ . For every  $i$ , let  $S_i \subseteq [n]$  denote the set whose characteristic vector is  $a_i$ , and observe that for every  $x \in \mathbb{F}_2^n$ ,

$$\mathbf{1}_V(x) = \prod_{i=1}^k \left( \frac{1 + (-1)^{b_i} \cdot \chi_{S_i}(x)}{2} \right).$$

This representation implies that  $\mathbf{1}_V$  is  $2^k$ -Fourier-sparse. ◀

### Chernoff-Hoeffding Bound

► **Theorem 2.5.** *Let  $X_1, \dots, X_N$  be  $N$  identically distributed independent random variables in  $[-a, +a]$  satisfying  $\mathbb{E}[X_i] = \mu$  for all  $i$ . Then for every  $\delta \leq 1/2$  and  $N \geq C \cdot a^2 \cdot \log(1/\delta)/\varepsilon^2$ , for a universal constant  $C$ , it holds that*

$$\Pr \left[ \left| \mu - \frac{1}{N} \cdot \sum_{i=1}^N X_i \right| < \varepsilon \right] \geq 1 - \delta.$$

## 3 The List-Decoding Size of Fourier-Sparse Boolean Functions

We turn to prove Theorem 1.1, which provides an upper bound on the list-decoding size of the code of block length  $2^n$  of all  $k$ -Fourier-sparse Boolean functions on  $n$  variables. Equivalently, for a general distance  $d$  and a function  $f : \mathbb{F}_2^n \rightarrow \mathbb{R}$  we bound the number of  $k$ -Fourier-sparse Boolean functions on  $n$  variables of distance at most  $d$  from  $f$ .

We start by proving that a function  $f : \mathbb{F}_2^n \rightarrow \mathbb{R}$  with small spectral norm can be well approximated by a linear combination of few functions from  $\{\chi_S\}_{S \subseteq [n]}$  with coefficients of equal magnitude. This was essentially proved in [9] and we include here the proof for completeness.

► **Lemma 3.1.** *For every function  $f : \mathbb{F}_2^n \rightarrow \mathbb{R}$ ,  $\varepsilon > 0$ , and  $\delta \in (0, 1/2]$ , there exists a collection<sup>2</sup>  $\mathcal{F}$  of  $O(\|\hat{f}\|_1^2 \cdot \log(1/\delta)/\varepsilon^2)$  subsets of  $[n]$  with signs  $(a_S \in \{\pm 1\})_{S \in \mathcal{F}}$  such that for all but at most  $\delta$  fraction of  $x \in \mathbb{F}_2^n$  it holds that*

$$\left| f(x) - \frac{\|\hat{f}\|_1}{|\mathcal{F}|} \cdot \sum_{S \in \mathcal{F}} a_S \cdot \chi_S(x) \right| < \varepsilon.$$

<sup>2</sup> Repetitions of subsets in the collection  $\mathcal{F}$  are allowed.

**Proof.** Observe that the function  $f$  can be represented as follows.

$$f = \sum_{S \subseteq [n]} \hat{f}(S) \cdot \chi_S = \sum_{S \subseteq [n]} \frac{|\hat{f}(S)|}{\|\hat{f}\|_1} \cdot \|\hat{f}\|_1 \cdot \text{sign}(\hat{f}(S)) \cdot \chi_S = \mathbb{E}_{S \sim D} [\|\hat{f}\|_1 \cdot \text{sign}(\hat{f}(S)) \cdot \chi_S],$$

where  $D$  is the distribution defined by  $D(S) = |\hat{f}(S)|/\|\hat{f}\|_1$ . Let  $\mathcal{F}$  be a collection of  $|\mathcal{F}| = O(\|\hat{f}\|_1^2 \cdot \log(1/\delta)/\varepsilon^2)$  independent random samples from the distribution  $D$ . For every  $x \in \mathbb{F}_2^n$ , the Chernoff-Hoeffding bound (Theorem 2.5) implies that with probability at least  $1 - \delta$  it holds that

$$\left| f(x) - \frac{1}{|\mathcal{F}|} \cdot \sum_{S \in \mathcal{F}} \|\hat{f}\|_1 \cdot a_S \cdot \chi_S(x) \right| < \varepsilon, \tag{2}$$

where  $a_S = \text{sign}(\hat{f}(S))$ . By linearity of expectation, it follows that there exist  $\mathcal{F}$  and signs  $(a_S)_{S \in \mathcal{F}}$  for which (2) holds for all but at most  $\delta$  fraction of  $x \in \mathbb{F}_2^n$ , as required. ◀

We now apply Lemma 3.1 to Fourier-sparse functions in  $\mathbb{F}_2^n \rightarrow \{-1, 0, +1\}$  with bounded support size, and then, in Corollary 3.3, derive an upper bound on the number of these functions.

► **Corollary 3.2.** *Let  $f : \mathbb{F}_2^n \rightarrow \{-1, 0, +1\}$  be a  $k$ -Fourier-sparse function satisfying  $|\text{supp}(f)| \leq d$ . Then for every  $\delta \in (0, 1/2]$  there exists a collection  $\mathcal{F}$  of  $O(dk \log(1/\delta)/2^n)$  subsets of  $[n]$  with signs  $(a_S \in \{\pm 1\})_{S \in \mathcal{F}}$  such that for all but at most  $\delta$  fraction of  $x \in \mathbb{F}_2^n$  it holds that*

$$\left| f(x) - \frac{\|\hat{f}\|_1}{|\mathcal{F}|} \cdot \sum_{S \in \mathcal{F}} a_S \cdot \chi_S(x) \right| < \frac{1}{2}.$$

**Proof.** By the Cauchy-Schwarz inequality and Parseval's theorem, we obtain that

$$\frac{\|\hat{f}\|_1^2}{k} \leq \sum_{S \subseteq [n]} \hat{f}(S)^2 = 2^{-n} \cdot \sum_{x \in \mathbb{F}_2^n} f(x)^2 \leq \frac{d}{2^n}.$$

The corollary follows from Lemma 3.1, applied with  $\varepsilon = 1/2$ , for  $|\mathcal{F}| = O(\|\hat{f}\|_1^2 \log(1/\delta)/\varepsilon^2) = O(dk \log(1/\delta)/2^n)$ . ◀

► **Corollary 3.3.** *The number of  $k$ -Fourier-sparse functions  $f : \mathbb{F}_2^n \rightarrow \{-1, 0, +1\}$  satisfying  $|\text{supp}(f)| \leq d$  is  $2^{O(ndk \log k/2^n)}$ .*

**Proof.** For every  $k$ -Fourier-sparse function  $f : \mathbb{F}_2^n \rightarrow \{-1, 0, +1\}$  satisfying  $|\text{supp}(f)| \leq d$ , let  $\mathcal{F}$  and  $(a_S)_{S \in \mathcal{F}}$  be as given by Corollary 3.2 for, say,  $\delta = 1/(5k)$ . Since the range of  $f$  is  $\{-1, 0, +1\}$ , it follows that the collection  $\mathcal{F}$ , the signs  $(a_S)_{S \in \mathcal{F}}$ , and the value of  $\|\hat{f}\|_1$  define a function of distance at most  $\delta \cdot 2^n$  from  $f$ . Notice that by Claim 2.2 and our choice of  $\delta$ , the distance between every two distinct  $k$ -Fourier-sparse functions is larger than  $2\delta \cdot 2^n$ . Thus, a function of distance at most  $\delta \cdot 2^n$  from  $f$  fully defines  $f$ . This implies that  $f$  can be represented by a binary string of length  $O(n \cdot dk \log k/2^n)$ , so the total number of such functions is  $2^{O(ndk \log k/2^n)}$ . ◀

The bound in Corollary 3.3 implies a bound on the number of Fourier-sparse Boolean functions of bounded distance from a given Boolean function.

► **Corollary 3.4.** *For every  $k$ -Fourier-sparse Boolean function  $f : \mathbb{F}_2^n \rightarrow \{0, 1\}$ , the number of  $k$ -Fourier-sparse Boolean functions of distance at most  $d$  from  $f$  is  $2^{O(ndk \log k/2^n)}$ .*

**Proof.** Let  $f : \mathbb{F}_2^n \rightarrow \{0, 1\}$  be a  $k$ -Fourier-sparse Boolean function. Consider the mapping that maps every  $k$ -Fourier-sparse Boolean function  $g : \mathbb{F}_2^n \rightarrow \{0, 1\}$ , whose distance from  $f$  is at most  $d$ , to the function  $h = f - g$ . Observe that  $h$  is a  $2k$ -Fourier-sparse function from  $\mathbb{F}_2^n$  to  $\{-1, 0, +1\}$  satisfying  $|\text{supp}(h)| \leq d$ . By Corollary 3.3, the number of such functions  $h$  is bounded by  $2^{O(ndk \log k/2^n)}$ . Since the above mapping is bijective, this bound holds for the number of functions  $g$  as well.  $\blacktriangleleft$

Equipped with Corollary 3.3, we restate and prove Theorem 1.1.

► **Theorem 1.1.** *For every function  $f : \mathbb{F}_2^n \rightarrow \mathbb{R}$ , the number of  $k$ -Fourier-sparse Boolean functions of distance at most  $d$  from  $f$  is  $2^{O(ndk \log k/2^n)}$ .*

**Proof.** If there is no  $k$ -Fourier-sparse Boolean function of distance at most  $d$  from  $f$ , then the bound trivially holds. So assume that such a function  $g : \mathbb{F}_2^n \rightarrow \{0, 1\}$  exists. Observe that every  $k$ -Fourier-sparse Boolean function of distance at most  $d$  from  $f$  has distance at most  $2d$  from  $g$ . Thus, by Corollary 3.4 applied to  $g$ , the number of such functions is at most  $2^{O(ndk \log k/2^n)}$ .  $\blacktriangleleft$

### 3.1 The Sample Complexity of Learning Fourier-Sparse Boolean Functions

The *sample complexity* of learning a class of functions is the minimum number of uniform and independent random samples needed from a function in the class for specifying it with high success probability. Here we consider the class of  $k$ -Fourier-sparse Boolean functions on  $n$  variables, and show how Theorem 1.1 implies an upper bound on the sample complexity of learning it (Corollary 3.6).

► **Theorem 3.5.** *For every  $n$ ,  $1 < k \leq 2^n$ , and a  $k$ -Fourier-sparse function  $f : \mathbb{F}_2^n \rightarrow \mathbb{R}$ , the following holds. The probability that when sampling  $O(n \cdot k \log k)$  uniform and independent random samples from  $f$ , there exists a  $k$ -Fourier-sparse Boolean function  $g \neq f$  that agrees with  $f$  on all the samples is  $2^{-\Omega(n \log k)}$ .*

**Proof.** Consider  $q = O(nk \log k)$  samples  $(x, f(x))$  from a  $k$ -Fourier-sparse function  $f : \mathbb{F}_2^n \rightarrow \mathbb{R}$ , where  $x$  is distributed uniformly and independently in  $\mathbb{F}_2^n$ . By Claim 2.2, the distance between  $f$  and every other  $k$ -Fourier-sparse function is at least  $2^n/(2k)$ . For an integer  $\ell \in [1, \lceil \log_2 2k \rceil]$ , consider all the  $k$ -Fourier-sparse Boolean functions whose distance from  $f$  is in  $[2^{n-\ell}, 2^{n-\ell+1}]$ . By Theorem 1.1, the number of such functions is  $2^{O(nk \log k/2^\ell)}$ . The probability that such a function agrees with  $q$  random independent samples of  $f$  is at most  $(1 - 2^{-\ell})^q$ . By the union bound, the probability that at least one of these functions agrees with the  $q$  samples is at most

$$2^{O(nk \log k/2^\ell)} \cdot (1 - 2^{-\ell})^q \leq 2^{O(nk \log k/2^\ell)} \cdot e^{-q/2^\ell} \leq 2^{-\Omega(n \log k)},$$

where the last inequality holds for an appropriate choice of  $q = O(nk \log k)$ . By applying the union bound over all the values of  $\ell$ , it follows that with probability  $1 - 2^{-\Omega(n \log k)}$  all the  $k$ -Fourier-sparse Boolean functions (besides  $f$ ) are eliminated, completing the proof.  $\blacktriangleleft$

The following corollary follows immediately from Theorem 3.5 and confirms Corollary 1.2.

► **Corollary 3.6.** *For every  $n$  and  $1 \leq k \leq 2^n$ , the number of uniform and independent random samples required for learning the class of  $k$ -Fourier-sparse Boolean functions on  $n$  variables with success probability  $1 - 2^{-\Omega(n \log k)}$  is  $O(n \cdot k \log k)$ .*

We end with the following simple lower bound.

► **Theorem 3.7.** *For every  $n$  and  $1 \leq k \leq 2^n$ , the number of uniform and independent random samples required for learning the class of  $k$ -Fourier-sparse Boolean functions on  $n$  variables with constant success probability is  $\Omega(k \cdot (n - \log_2 k))$ .*

**Proof.** Assume without loss of generality that  $k$  is a power of 2. Let  $A$  be an algorithm for learning the class above with constant success probability  $p > 0$  using  $q$  uniform and independent random samples. Consider the class  $\mathcal{G}$  of indicators of affine subspaces of  $\mathbb{F}_2^n$  of co-dimension  $\log_2 k$  (i.e., affine subspaces of  $\mathbb{F}_2^n$  of size  $2^n/k$ ). By Claim 2.4, the functions in  $\mathcal{G}$  are  $k$ -Fourier-sparse. Observe that their number satisfies

$$|\mathcal{G}| = 2^{\Theta(n \cdot \min(\log_2 k, n - \log_2 k))}.$$

By Yao's minimax principle, there exists a deterministic algorithm  $A'$  (obtained by fixing the random coins of  $A$ ) that given evaluations of a function, chosen uniformly at random from  $\mathcal{G}$ , on a *fixed* collection of  $q$  points in  $\mathbb{F}_2^n$ , learns it with success probability  $p$ .

Now, observe that the expected number of 1-evaluations that  $A'$  receives is  $q/k$ . By Markov's inequality, the probability that  $A'$  receives at least  $2q/(pk)$  1-evaluations is at most  $p/2$ . It follows that for at least  $p/2$  fraction of the functions in  $\mathcal{G}$  the algorithm  $A'$  receives at most  $2q/(pk)$  1-evaluations and learns them correctly. Assuming that  $pk \geq 2$ , the number of possible evaluation sequences on these inputs is at most

$$\sum_{i=0}^{2q/(pk)} \binom{q}{i} \leq (k \cdot pe/2)^{2q/(pk)} \leq 2^{O(q \cdot \log_2 k/k)},$$

where for the first inequality we used the standard inequality  $\sum_{i=0}^t \binom{q}{i} \leq (qe/t)^t$  which holds for  $t \leq q$  (see, e.g., [16, Proposition 1.4]). The above is bounded from below by  $|\mathcal{G}| \cdot p/2$ , implying that

$$q \geq \Omega(n \cdot \min(\log_2 k, n - \log_2 k) \cdot k / \log_2 k) \geq \Omega(k \cdot (n - \log_2 k)),$$

where the last inequality follows by considering separately the cases of  $k \geq 2^{n/2}$  and  $k < 2^{n/2}$ . In case that  $pk < 2$ , the number of possible evaluation sequences is at most  $2^q$ , and the bound follows similarly using the assumption that  $p$  is a fixed constant. ◀

## 4 Testing Booleanity of Fourier-Sparse Functions

In this section we prove upper and lower bounds on the query complexity of testing Booleanity of Fourier-sparse functions. For a parameter  $k$ , consider the problem in which given access to a  $k$ -Fourier-sparse function  $f : \mathbb{F}_2^n \rightarrow \mathbb{R}$  one has to decide if  $f$  is Boolean, i.e.,  $f(x) \in \{0, 1\}$  for every  $x \in \mathbb{F}_2^n$ , or not, with some constant success probability.

### 4.1 Upper Bound

As mentioned before, Gur and Tamuz proved in [14] that every  $k$ -Fourier-sparse non-Boolean function  $f$  on  $n$  variables satisfies  $f(x) \notin \{0, 1\}$  for at least  $\Omega(2^n/k^2)$  inputs  $x \in \mathbb{F}_2^n$  (see Claim 2.3). Thus, querying the input function  $f$  on  $O(k^2)$  independent and random inputs suffices in order to catch a non-Boolean value of  $f$  if such a value exists. In the following lemma it is shown that it is not really needed to choose the  $O(k^2)$  random vectors *independently*. It turns out that a restriction of a  $k$ -Fourier-sparse non-Boolean function to a random linear

subspace of size  $O(k^2)$ , that is, of dimension  $\approx 2 \log_2 k$ , is with high probability non-Boolean. Thus, the tester could randomly pick such a subspace and query  $f$  on all of its vectors. This decreases the amount of randomness used in the tester of [14] from  $O(nk^2)$  to  $O(n \log k)$ . More importantly for us, this reduces the problem of testing Booleanity of  $k$ -Fourier-sparse functions on  $n$  variables to the case of  $k = \Theta(2^{n/2})$ .

► **Lemma 4.1.** *Let  $f : \mathbb{F}_2^n \rightarrow \mathbb{R}$  be a  $k$ -Fourier-sparse non-Boolean function, and denote  $L = (k^2 + k + 2)/2$ . Then, for every  $\delta > 0$ , the restriction of  $f$  to a uniformly chosen random linear subspace of dimension  $r \geq \log_2(L/\delta)$  is also non-Boolean with probability at least  $1 - \delta$ .*

**Proof.** Let  $f : \mathbb{F}_2^n \rightarrow \mathbb{R}$  be a  $k$ -Fourier-sparse non-Boolean function. By Claim 2.3, there are at least  $2^n/L$  vectors  $x \in \mathbb{F}_2^n$  for which  $f(x) \notin \{0, 1\}$ . This implies that there exists a set  $S$  of at least  $\log_2(2^n/L)$  linearly independent vectors in  $\mathbb{F}_2^n$  on which  $f$  is not Boolean. Consider a linear subspace  $V \subseteq \mathbb{F}_2^n$  of dimension  $n - 1$  chosen uniformly at random. Since the vectors in  $S$  are linearly independent, the probability that no vector in  $S$  is in  $V$  is  $2^{-|S|} \leq \frac{L}{2^n}$ . It follows that the restriction  $f|_V$  of  $f$  to  $V$  is a  $k$ -Fourier-sparse function defined on a linear subspace of dimension  $n - 1$ , and its probability to be Boolean is at most  $\frac{L}{2^n}$ . Note that one can think of the domain of  $f|_V$  as  $\mathbb{F}_2^{n-1}$ , because  $V$  and  $\mathbb{F}_2^{n-1}$  are isomorphic and a composition with an invertible linear transformation does not affect the Fourier-sparsity. Now, let us repeat the above process  $n - r - 1$  additional times, until we get a linear subspace of dimension  $r$ . The probability that the function becomes Boolean in one of the steps is at most

$$\frac{L}{2^n} + \frac{L}{2^{n-1}} + \cdots + \frac{L}{2^{r+1}} \leq \frac{L}{2^r} \leq \delta,$$

and we are done. ◀

We now restate and prove Theorem 1.3, which gives an upper bound of  $O(k \cdot \log^2 k)$  on the query complexity of testing Booleanity of  $k$ -Fourier-sparse functions. In the proof, we first apply Lemma 4.1 to restrict the input function to a subspace of dimension  $O(\log k)$ . Then, we apply Theorem 3.5 in an attempt to learn the restricted function and check if it is consistent with some  $k$ -Fourier-sparse Boolean function.

► **Theorem 1.3.** *For every  $k$  there exists a non-adaptive one-sided error tester that using  $O(k \cdot \log^2 k)$  queries to an input  $k$ -Fourier-sparse function  $f : \mathbb{F}_2^n \rightarrow \mathbb{R}$  decides if  $f$  is Boolean or not with constant success probability.*

**Proof.** Consider the tester that given access to an input  $k$ -Fourier-sparse function  $f : \mathbb{F}_2^n \rightarrow \mathbb{R}$  acts as follows:

1. Pick uniformly at random a linear subspace  $V$  of  $\mathbb{F}_2^n$  of dimension  $r = \min(n, \lceil \log_2(100L) \rceil)$ , where  $L = (k^2 + k + 2)/2$ , and let  $T$  be an invertible linear transformation mapping  $\mathbb{F}_2^r$  to  $V$ .
2. Query  $f$  on  $O(r \cdot k \log k)$  random vectors chosen uniformly and independently from the subspace  $V$ . Note that these queries can be seen as uniform and independent random samples from the function  $g : \mathbb{F}_2^r \rightarrow \mathbb{R}$  defined as  $g = f \circ T$ .
3. If there exists a  $k$ -Fourier-sparse Boolean function on  $r$  variables that agrees with the above samples of  $g$  then accept, and otherwise reject.

We turn to prove the correctness of the above tester. If  $f$  is a  $k$ -Fourier-sparse Boolean function then so is  $g$ , because a restriction to a subspace and a composition with a linear transformation leave the function  $k$ -Fourier-sparse and Boolean. Hence, in this case the tester accepts with probability 1.

On the other hand, if  $f$  is a  $k$ -Fourier-sparse non-Boolean function, then by Lemma 4.1 the restriction of  $f$  to the random subspace  $V$  of dimension  $r$  picked in Item 1, as well as the function  $g$  defined in Item 2, are also non-Boolean with probability at least 0.99. In this case, by Theorem 3.5, the probability that there is a  $k$ -Fourier-sparse Boolean function on  $r$  variables that agrees with  $O(r \cdot k \log k)$  uniform and independent random samples from  $g$  is  $2^{-\Omega(r \log k)}$ , thus the tester correctly rejects with probability at least, say, 0.9, as required. Finally, observe that the number of queries made by the tester is  $O(r \cdot k \log k) = O(k \cdot \log^2 k)$ . ◀

## 4.2 Lower Bound

We turn to restate and prove our lower bound on the query complexity of testing Booleanity of  $k$ -Fourier-sparse functions.

► **Theorem 1.4.** *Every non-adaptive one-sided error tester for Booleanity of  $k$ -Fourier-sparse functions has query complexity  $\Omega(k \cdot \log k)$ .*

**Proof.** For a given integer  $k$ , let  $n$  be the largest even integer satisfying  $k \geq 3 \cdot 2^{n/2}$ . Define a distribution  $D_{no}$  over functions in  $\mathbb{F}_2^n \rightarrow \{0, 1, 2\}$  as follows. Pick uniformly at random a pair  $(V_1, V_2)$  of affine subspaces satisfying  $\dim(V_1) = \dim(V_2) = n/2$  and  $|V_1 \cap V_2| = 1$ , and output the sum of indicators  $\mathbf{1}_{V_1} + \mathbf{1}_{V_2}$ . Notice that, by Claim 2.4, such a function has Fourier-sparsity at most  $2 \cdot 2^{n/2} \leq k$ . Thus, a function chosen from  $D_{no}$  is  $k$ -Fourier-sparse and non-Boolean with probability 1.

Let  $T$  be a non-adaptive one-sided error randomized tester for Booleanity of  $k$ -Fourier-sparse functions with query complexity  $q$  and success probability at least  $2/3$ . By Yao's minimax principle, there exists a deterministic tester  $T'$  (obtained by fixing the random coins of  $T$ ) that rejects a random function chosen from  $D_{no}$  with probability at least  $2/3$ . Since  $T$  is non-adaptive and has one-sided error, it follows that  $T'$  queries an input function on  $q$  fixed vectors  $a_1, \dots, a_q \in \mathbb{F}_2^n$ , accepts every  $k$ -Fourier-sparse Boolean function, and rejects a function chosen from  $D_{no}$  with probability at least  $2/3$ . We turn to prove that  $q > (n \cdot 2^{n/2})/1000 = \Omega(k \cdot \log k)$ .

Assume in contradiction that  $q \leq (n \cdot 2^{n/2})/1000$ . Let  $f$  be a random function chosen from  $D_{no}$ , that is,  $f = \mathbf{1}_{V_1} + \mathbf{1}_{V_2}$  for random affine subspaces  $V_1$  and  $V_2$  of dimension  $n/2$  satisfying  $|V_1 \cap V_2| = 1$ . For  $i = 1, 2$ , let  $W_i$  be the affine span of  $\{a_1, \dots, a_q\} \cap V_i$ . Let  $E$  be the event that the intersection of  $W_1$  and  $W_2$  is empty. We turn to prove that if the event  $E$  happens then the tester  $T'$  accepts the function  $f$  and that the probability of this event is at least 0.9. This contradicts the success probability of  $T'$  on functions chosen from  $D_{no}$  and completes the proof.

► **Lemma 4.2.** *If the event  $E$  happens then the tester  $T'$  accepts the function  $f$ .*

**Proof.** Assume that the event  $E$  happens, i.e.,  $W_1 \cap W_2 = \emptyset$ . Then, there exists an affine subspace  $V'_2$  of dimension  $n/2 - 1$  satisfying  $W_2 \subseteq V'_2 \subsetneq V_2$  and  $V_1 \cap V'_2 = \emptyset$ . Consider the function  $g = \mathbf{1}_{V_1} + \mathbf{1}_{V'_2}$ . By Claim 2.4,  $g$  is a Boolean function whose Fourier-sparsity is at most  $3 \cdot 2^{n/2} \leq k$ , thus it is accepted by  $T'$ . However,  $g$  satisfies  $g(a_i) = f(a_i)$  for every  $1 \leq i \leq q$ . This implies that  $T'$  cannot distinguish between  $g$  and  $f$ , so it must accept  $f$  as well. ◀

► **Lemma 4.3.** *The probability of the event  $E$  is at least 0.9.*



**Proof.** Denote by  $X$  the number of vectors in  $\{a_1, \dots, a_q\} \cap V_1$ . Since  $V_1$  is distributed uniformly over all affine subspaces of dimension  $n/2$ , the probability that  $a_i$  belongs to  $V_1$  is  $2^{-n/2}$  for every  $1 \leq i \leq q$ . Thus, by linearity of expectation,

$$\mathbb{E}[X] = \frac{q}{2^{n/2}} \leq \frac{(n \cdot 2^{n/2})/1000}{2^{n/2}} = \frac{n}{1000}.$$

By Markov's inequality, we obtain that

$$\Pr \left[ \dim(W_1) \geq \frac{n}{10} \right] \leq \Pr \left[ X \geq \frac{n}{10} \right] \leq \frac{1}{100}.$$

Now, fix a choice of  $V_1$  for which  $\dim(W_1) < n/10$ , and consider the randomness over the choice of  $V_2$ . Notice that, conditioned on  $V_1$ ,  $V_2$  is distributed uniformly over all the affine subspaces of dimension  $n/2$  which contain exactly one vector from  $V_1$ . By symmetry, every vector of  $V_1$  has probability  $|V_1|^{-1} = 2^{-n/2}$  to belong to  $V_2$ . Thus, the probability that the vector that belongs to both  $V_1$  and  $V_2$  is in  $W_1$  is  $|W_1| \cdot 2^{-n/2} < 2^{n/10} \cdot 2^{-n/2} = 2^{-2n/5}$ .

Finally, the probability that  $W_1 \cap W_2 = \emptyset$  is at least the probability that  $W_1 \cap V_2 = \emptyset$ , and the latter is at least  $1 - (0.01 + 2^{-2n/5}) \geq 0.9$  for every sufficiently large  $n$ .  $\blacktriangleleft$

$\blacktriangleleft$

**Acknowledgements.** We thank Adi Akavia, Shachar Lovett, and Eric Price for useful discussions and comments.

---

## References

- 1 Noga Alon, Tali Kaufman, Michael Krivelevich, Simon Litsyn, and Dana Ron. Testing Reed-Muller codes. *IEEE Trans. on Information Theory*, 51(11):4032–4039, 2005. Preliminary version in RANDOM'03.
- 2 Alexandr Andoni, Rina Panigrahy, Gregory Valiant, and Li Zhang. Learning sparse polynomial functions. In *SODA*, pages 500–510, 2014.
- 3 Khanh Do Ba, Piotr Indyk, Eric Price, and David P. Woodruff. Lower bounds for sparse recovery. In *SODA*, pages 1190–1197, 2010.
- 4 Anna Bernasconi and Bruno Codenotti. Spectral analysis of Boolean functions as a graph eigenvalue problem. *IEEE Trans. on Computers*, 48(3):345–351, 1999.
- 5 Arnab Bhattacharyya. Guest column: On testing affine-invariant properties over finite fields. *SIGACT News*, 44(4):53–72, 2013.
- 6 Arnab Bhattacharyya, Swastik Kopparty, Grant Schoenebeck, Madhu Sudan, and David Zuckerman. Optimal testing of Reed-Muller codes. In *FOCS*, pages 488–497, 2010.
- 7 Avrim Blum. Learning a function of  $r$  relevant variables. In *COLT*, pages 731–733, 2003.
- 8 Jean Bourgain. An improved estimate in the restricted isometry problem. In *Geometric Aspects of Functional Analysis*, volume 2116 of *Lecture Notes in Mathematics*, pages 65–70. Springer, 2014.
- 9 Jehoshua Bruck and Roman Smolensky. Polynomial threshold functions,  $AC^0$  functions, and spectral norms. *SIAM J. Comput.*, 21(1):33–42, 1992. Preliminary version in FOCS'90.
- 10 Emmanuel J. Candès and Terence Tao. Near-optimal signal recovery from random projections: Universal encoding strategies? *IEEE Trans. on Information Theory*, 52(12):5406–5425, 2006.
- 11 Mahdi Cheraghchi, Venkatesan Guruswami, and Ameya Velingker. Restricted isometry of Fourier matrices and list decodability of random linear codes. *SIAM J. Comput.*, 42(5):1888–1914, 2013. Preliminary version in SODA'13.



- 12 Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In *STOC*, pages 25–32, 1989.
- 13 Parikshit Gopalan, Ryan O’Donnell, Rocco A. Servedio, Amir Shpilka, and Karl Wimmer. Testing Fourier dimensionality and sparsity. *SIAM J. Comput.*, 40(4):1075–1100, 2011. Preliminary version in ICALP’09.
- 14 Tom Gur and Omer Tamuz. Testing Booleanity and the uncertainty principle. *Chicago J. Theor. Comput. Sci.*, 2013, 2013.
- 15 Ishay Haviv and Oded Regev. The restricted isometry property of subsampled Fourier matrices, 2015. Manuscript.
- 16 Stasys Jukna. *Extremal Combinatorics: With Applications in Computer Science*. Texts in theoretical computer science. Springer-Verlag, second edition, 2011.
- 17 Tali Kaufman, Shachar Lovett, and Ely Porat. Weight distribution and list-decoding size of Reed-Muller codes. *IEEE Trans. on Information Theory*, 58(5):2689–2696, 2012. Preliminary version in ICS’10.
- 18 Murat Kocaoglu, Karthikeyan Shanmugam, Alexandros G. Dimakis, and Adam R. Klivans. Sparse polynomial learning and graph sketching. In *NIPS*, pages 3122–3130, 2014.
- 19 Eyal Kushilevitz and Yishay Mansour. Learning decision trees using the Fourier spectrum. *SIAM J. Comput.*, 22(6):1331–1348, 1993. Preliminary version in STOC’91.
- 20 Nathan Linial, Yishay Mansour, and Noam Nisan. Constant depth circuits, Fourier transform, and learnability. *J. ACM*, 40(3):607–620, 1993. Preliminary version in FOCS’89.
- 21 Elchanan Mossel, Ryan O’Donnell, and Rocco A. Servedio. Learning functions of  $k$  relevant variables. *J. Comput. Syst. Sci.*, 69(3):421–434, 2004. Preliminary version in STOC’03.
- 22 Ryan O’Donnell. *Analysis of Boolean Functions*. Cambridge University Press, 2014.
- 23 Mark Rudelson and Roman Vershynin. On sparse reconstruction from Fourier and Gaussian measurements. *Comm. Pure Appl. Math.*, 61(8):1025–1045, 2008.
- 24 Madhu Sudan. Invariance in property testing. In *Property Testing – Current Research and Surveys*, volume 6390, pages 211–227. Springer, 2010.
- 25 Gregory Valiant. Finding correlations in subquadratic time, with applications to learning parities and juntas. In *FOCS*, pages 11–20, 2012.

# Nonclassical Polynomials as a Barrier to Polynomial Lower Bounds

Abhishek Bhowmick\*<sup>1</sup> and Shachar Lovett†<sup>2</sup>

1 Department of Computer Science  
The University of Texas at Austin, USA  
bhowmick@cs.utexas.edu

2 Department of Computer Science and Engineering  
University of California, San Diego, USA  
slovett@ucsd.edu

---

## Abstract

The problem of constructing explicit functions which cannot be approximated by low degree polynomials has been extensively studied in computational complexity, motivated by applications in circuit lower bounds, pseudo-randomness, constructions of Ramsey graphs and locally decodable codes. Still, most of the known lower bounds become trivial for polynomials of super-logarithmic degree. Here, we suggest a new barrier explaining this phenomenon. We show that many of the existing lower bound proof techniques extend to nonclassical polynomials, an extension of classical polynomials which arose in higher order Fourier analysis. Moreover, these techniques are tight for nonclassical polynomials of logarithmic degree.

**1998 ACM Subject Classification** F.2.2 Nonnumerical Algorithms and Problems

**Keywords and phrases** nonclassical polynomials, polynomials, lower bounds, barrier

**Digital Object Identifier** 10.4230/LIPIcs.CCC.2015.72

## 1 Introduction

Polynomials play a fundamental role in computer science with important applications in algorithm design, coding theory, pseudo-randomness, cryptography and complexity theory. They are also instrumental in proving lower bounds, as many lower bounds techniques first reduce the computational model to a computation or an approximation by a low degree polynomial, and then continue to show that certain hard functions cannot be computed or approximated by low degree polynomials. Motivated by these applications, the problem of constructing explicit functions which cannot be computed or approximated (in certain ways) by low degree polynomials has been widely explored in computational complexity. However, most techniques to date apply only to relative low degree polynomials. In this paper, we focus on understanding this phenomenon, when the polynomials are defined over fixed size finite fields. In this regime, many lower bound techniques become trivial when the degree grows beyond logarithmic in the number of variables. We propose a new barrier explaining the lack of ability to prove strong lower bounds for polynomials of super-logarithmic degree. The barrier is based on *nonclassical polynomials*, an extension of standard (classical) polynomials which arose in higher order Fourier analysis. We show that several existing lower bound techniques extend to nonclassical polynomials, for which the logarithmic degree bound is

---

\* Research supported in part by NSF Grant CCF-1218723.

† Research supported in part by NSF CAREER award 1350481.



tight. Hence, to prove stronger lower bounds, one should either focus on techniques which distinguish classical from nonclassical polynomials, or consider functions which are hard also for nonclassical polynomials.

### 1.1 Nonclassical polynomials

Nonclassical polynomials were introduced by Tao and Ziegler [24] in their works on the inverse theorem for the Gowers uniformity norms. To introduce these, it will be beneficial to first consider classical polynomials. Fix a prime finite field  $\mathbb{F}_p$ , where we consider  $p$  to be a constant. A function  $f : \mathbb{F}_p^n \rightarrow \mathbb{F}_p$  is a degree  $d$  polynomial if it can be written as a linear combination of monomials of degree at most  $d$ . An equivalent definition is that  $f$  is annihilated by taking any  $d + 1$  directional derivatives. That is, for a direction  $h \in \mathbb{F}_p^n$  define the derivative of  $f$  in direction  $h$  as  $D_h f(x) = f(x + h) - f(x)$ . Then,  $f$  is a polynomial of degree at most  $d$  iff

$$D_{h_1} \dots D_{h_{d+1}} f \equiv 0 \quad \forall h_1, \dots, h_{d+1} \in \mathbb{F}_p^n.$$

Nonclassical polynomials extend this definition to a larger class of objects. Let  $\mathbb{T} = \mathbb{R}/\mathbb{Z}$  denote the torus. For a function  $f : \mathbb{F}_p^n \rightarrow \mathbb{T}$ , define its directional derivative in direction  $h \in \mathbb{F}_p^n$  as before, as  $D_h f(x) = f(x + h) - f(x)$ . Then, we define  $f$  to be a *nonclassical polynomial of degree at most  $d$*  if it is annihilated by any  $d + 1$  derivatives,

$$D_{h_1} \dots D_{h_{d+1}} f \equiv 0 \quad \forall h_1, \dots, h_{d+1} \in \mathbb{F}_p^n.$$

While not immediately obvious, the class of nonclassical polynomials contains the classical polynomials. Let  $|\cdot| : \mathbb{F}_p \rightarrow \{0, \dots, p-1\} \subset \mathbb{Z}$  denote the natural embedding. If  $f : \mathbb{F}_p^n \rightarrow \mathbb{F}_p$  is a classical polynomial of degree  $d$  then  $|f(x)|/p \pmod{1}$  is a nonclassical polynomial of degree  $d$ . It turns out that as long as  $d < p$ , these capture all the nonclassical polynomials. However, for  $d \geq p$  nonclassical polynomials strictly extend classical polynomials of the same degree. For example, the following is a nonclassical polynomial of degree  $p$ :

$$f(x) = \frac{\sum |x_i|}{p^2}.$$

See Section 2 for more details on nonclassical polynomials.

### 1.2 Correlation bounds for polynomials

We first consider the problem of constructing explicit boolean functions which cannot be approximated by low-degree polynomials. For simplicity, we focus on polynomials defined over  $\mathbb{F}_2$ , but note that the results below extend to any constant prime finite field. This problem was studied by Razborov [22] and Smolensky [23] in the context of proving lower bounds for  $\text{AC}^0(\oplus)$  circuits (and more generally, bounded depth circuits with modular gates modulo a fixed prime). Consider for example the function  $\text{MOD}_3 : \{0, 1\}^n \rightarrow \{0, 1\}$ , which outputs 1 if the sum of the bits is zero modulo 3, and outputs 0 otherwise. The probability that it outputs 0 is very close to  $2/3$ . They showed that low degree polynomials over  $\mathbb{F}_2$  cannot improve this significantly. If  $f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2$  be a polynomial of degree  $d$  then

$$\Pr_{x \in \{0,1\}^n} [f(x) = \text{MOD}_3(x)] \leq \frac{2}{3} + O\left(\frac{d}{\sqrt{n}}\right).$$

This is sufficient to prove that the  $\text{MOD}_3$  function cannot be computed by sub-exponential  $\text{AC}^0(\oplus)$  circuits. However, one would like to prove that it cannot even be slightly approximated by such circuits. Such a result would be a major step towards constructing

pseudorandom generators for  $\text{AC}^0(\oplus)$  circuits [20, 21], a well known open problem in circuit complexity. It turns out that the Razborov-Smolensky bound is tight for very large degrees, as there exist polynomials of degree  $d = \Omega(\sqrt{n})$  which approximate the  $\text{MOD}_3$  function with probability 0.99, say. However, it seems to be far from tight for  $d \ll \sqrt{n}$ , which suggests that an alternative proof technique may be needed.

Viola and Wigderson [26] proved stronger inapproximability results for degrees  $d \ll \log n$ . These are better described if one considers the correlation of  $f$  with the sum of the bits modulo 3. In the following, let  $\omega_3 = \exp(2\pi i/3)$  be a cubic root of unity. They showed that if  $f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2$  is a polynomial of degree  $d$  then

$$\mathbb{E}_{x \in \{0,1\}^n} \left[ (-1)^{f(x)} \omega_3^{x_1 + \dots + x_n} \right] \leq 2^{-\Omega(n/4^d)}.$$

The technique of [26] proves exponential correlation bounds for constant degrees, but decays quickly and becomes trivial at  $d = O(\log n)$ . Our first result is that this is because of a good reason. Their technique is based on derivatives, and hence extends to nonclassical polynomials. Moreover, it is tight for nonclassical polynomials. In the following, let  $e : \mathbb{T} \rightarrow \mathbb{C}^*$  be defined as  $e(x) = \exp(2\pi i x)$ .

► **Theorem 1.1** (Correlation bounds with modular sums for nonclassical polynomials (informal)). *Let  $f : \mathbb{F}_2^n \rightarrow \mathbb{T}$  be a nonclassical polynomial of degree  $d$ . Then*

$$\mathbb{E}_{x \in \{0,1\}^n} \left[ e(f(x)) \omega_3^{x_1 + \dots + x_n} \right] \leq 2^{-\Omega(n/4^d)}.$$

Moreover, for any  $\varepsilon > 0$  there exists a nonclassical polynomial  $f : \mathbb{F}_2^n \rightarrow \mathbb{T}$  of degree  $O(\log(n/\varepsilon))$  such that

$$\mathbb{E}_{x \in \{0,1\}^n} \left[ e(f(x)) \omega_3^{x_1 + \dots + x_n} \right] \geq 1 - \varepsilon.$$

So, the Viola-Wigderson technique is bounded for degrees smaller than  $O(\log n)$ , because it extends to nonclassical polynomials of that degree, for which it is tight. We note that the modulus 3 in Theorem 1.1 can be replaced with any fixed odd modulus.

Another boolean function which was shown by Razborov and Smolensky [22, 23] to be hard for  $\text{AC}^0(\oplus)$  circuits is the majority function  $\text{MAJ} : \mathbb{F}_2^n \rightarrow \mathbb{F}_2$ . The proof relies on the following key fact. If  $f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2$  is a degree  $d$  polynomial then

$$\Pr_{x \in \{0,1\}^n} [f(x) = \text{MAJ}(x)] \leq \frac{1}{2} + O\left(\frac{d}{\sqrt{n}}\right).$$

Equivalently, this can be presented as a correlation bound

$$\mathbb{E}_{x \in \{0,1\}^n} \left[ (-1)^{f(x)} (-1)^{\text{MAJ}(x)} \right] \leq O\left(\frac{d}{\sqrt{n}}\right).$$

This is known to be tight for degree  $d = 1$  (as say  $x_1$  has correlation  $\Omega(1/\sqrt{n})$  with the majority function) and also for  $d = \Omega(\sqrt{n})$ , since there exist polynomials of that degree which approximate well the majority function, or any symmetric function for that matter. However, it is not known if these bounds are tight for degrees  $1 \ll d \ll \sqrt{n}$ . We study this question for nonclassical polynomials. We show that there are nonclassical polynomials of degree  $O(\log n)$  with a constant correlation with the majority function.

► **Theorem 1.2** (Correlation bounds with majority for nonclassical polynomials (informal)). *There exists a nonclassical polynomial  $f : \mathbb{F}_2^n \rightarrow \mathbb{T}$  of degree  $O(\log n)$  such that*

$$\left| \mathbb{E} \left[ e(f(x)) (-1)^{\text{MAJ}(x)} \right] \right| \geq \Omega(1).$$

So, the Razborov-Smolensky technique separates classical from nonclassical polynomials, since classical polynomials of degree  $O(\log n)$  have negligible correlation with the majority function, while as we show above, this is false for nonclassical polynomials.

### 1.3 Exact computation by polynomials

A related problem to correlation bounds is that of exact computation with good probability. For classical polynomials the two problems are equivalent, but this is not the case for nonclassical polynomials. Given a nonclassical polynomial  $f : \mathbb{F}_2^n \rightarrow \mathbb{T}$ , we can ask what is the probability that  $f$  is equal to a boolean function, say the majority function. To do so, we identify naturally  $\mathbb{F}_2$  with  $\{0, 1/2\} \subset \mathbb{T}$ , and consider  $\text{MAJ} : \mathbb{F}_2^n \rightarrow \{0, 1/2\}$ . We show the following result, which gives a partial answer to the question.

► **Theorem 1.3** (Exact computation of majority by nonclassical polynomials (informal)). *Let  $f : \mathbb{F}_2^n \rightarrow \mathbb{T}$  be a nonclassical polynomial of degree  $d$ . Then,*

$$\Pr_{x \in \{0,1\}^n} [f(x) = \text{MAJ}(x)] \leq \frac{1}{2} + O\left(\frac{d2^d}{\sqrt{n}}\right).$$

We believe that the bound is not tight, and that, unlike for correlation bounds, nonclassical polynomials should not be able to exactly compute boolean functions better than classical polynomials. Specifically, we ask the following question.

► **Open Problem 1.4.** *Let  $f : \mathbb{F}_2^n \rightarrow \mathbb{T}$  be a nonclassical polynomial of degree  $d$ . Show that*

$$\Pr_{x \in \{0,1\}^n} [f(x) = \text{MAJ}(x)] \leq \frac{1}{2} + O\left(\frac{d}{\sqrt{n}}\right).$$

### 1.4 Weak representation of the OR function

We next move to the problem of weak representation of the OR function. Let  $p_1, \dots, p_r$  be distinct primes and let  $m = p_1 \cdots p_r$ . The goal is to construct a low degree polynomial  $f \in \mathbb{Z}_m[x_1, \dots, x_n]$  such that  $f(0^n) = 0$  but  $f(x) \neq 0$  for all nonzero  $x \in \{0, 1\}^n$ . Such polynomials stand at the core of some of the best constructions of Ramsey graphs [13, 14, 16]<sup>1</sup> and locally decodable codes [8, 10–12, 27], and were further investigated in [3–7, 23]. There are currently exponential gaps between the best constructions and lower bounds. Barrington, Beigel and Rudich [5] showed that there exist polynomials of degree  $O(n^{1/r})$  that weakly represent the OR function. The best lower bound is  $\Omega(\log^{1/(r-1)} n)$ , due to Barrington and Tardos [3].

The definition of weak representation can be equivalently defined (via the Chinese Remainder Theorem) as follows. There exist polynomials  $f_i : \mathbb{F}_{p_i}^n \rightarrow \mathbb{F}_{p_i}$  for  $i = 1, \dots, r$  such that  $f_1(0^n) = \dots = f_r(0^n) = 0$  but for any nonzero  $x \in \{0, 1\}^n$ , there exists an  $i$  for which  $f_i(x) \neq 0$ . This definition can be naturally extended to nonclassical polynomials, where we consider  $f_i : \mathbb{F}_{p_i}^n \rightarrow \mathbb{T}$ . We show that the Barrington-Tardos lower bound extends to nonclassical polynomials, and it is tight up to polynomial factors.

► **Theorem 1.5** (Weak representation of OR for nonclassical polynomials (informal)). *Let  $p_1, \dots, p_r$  be distinct primes, and  $f_i : \mathbb{F}_{p_i}^n \rightarrow \mathbb{T}$  be nonclassical polynomials which weakly represent the OR function. Then*

$$\max \deg(f_i) \geq \Omega(\log^{1/r} n).$$

<sup>1</sup> The current record is due to [2] which uses different techniques.

Moreover, for any fixed prime  $p$ , there exists a nonclassical polynomial  $f : \mathbb{F}_p^n \rightarrow \mathbb{T}$  of degree  $O(\log n)$  which weakly represents the OR function.

Thus, the proof technique of Barrington-Tardos cannot extend beyond degree  $O(\log n)$ , as it applies to nonclassical polynomials as well, for which the  $O(\log n)$  bound holds even for prime modulus. We note that unlike in the case of Theorem 1.1, where the lower bound proof of [26] extended naturally to nonclassical polynomials, extending the lower bound technique of [3] to nonclassical polynomials requires several nontrivial modifications of the original proof.

As an aside, in the classical setting, we present an improvement in the degree of a symmetric polynomial that weakly represents OR. This improves the result in [5] in the growing modulus case and constructs a polynomial whose degree is modulus independent. For more details, see Appendix A.

### 1.5 Pseudorandom generators for low degree polynomials

Consider for simplicity polynomials over  $\mathbb{F}_2$ . A distribution  $D$  over  $\mathbb{F}_2^n$  is said to fool polynomials of degree  $d$  with error  $\varepsilon$ , if for any polynomial  $f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2$  of degree at most  $d$ , we have

$$|\Pr_{x \sim D}[f(x) = 0] - \Pr_{x \in \mathbb{F}_2^n}[f(x) = 0]| \leq \varepsilon.$$

Distributions which fool linear functions (e.g.  $d = 1$ ) are called small bias generators, and optimal constructions of them (up to polynomial factors) were given in [1, 19], with seed length  $O(\log n/\varepsilon)$ . A sequence of works [9, 17, 25] showed that small bias generators can be combined to yield generators for larger degree polynomials. The best construction to date is by Viola [25], who showed that the sum of  $d$  independent small bias generators with error approximately  $\varepsilon^{2^d}$  fools degree  $d$  polynomials with error  $\varepsilon$ . Thus, his construction has seed length  $O(2^d \log(1/\varepsilon) + d \log n)$ , and becomes trivial for  $d = \Omega(\log n)$ . It is not clear whether it is necessary to require the small bias generators to have smaller error than the required error for the degree  $d$  polynomials, and this is the main source for the loss in parameters when considering large degrees.

There is a natural extension of these definitions to nonclassical polynomials. If  $f : \mathbb{F}_2^n \rightarrow \mathbb{T}$  is a nonclassical polynomial of degree  $d$ , then we require that

$$|\mathbb{E}_{x \sim D}[e(f(x))] - \mathbb{E}_{x \in \mathbb{F}_2^n}[e(f(x))]| \leq \varepsilon.$$

The proof technique of Viola is based on derivatives, and we note here (without proof) that it extends to nonclassical polynomials in a straightforward way. We suspect that it is tight for nonclassical polynomials, however we were unable to show that. Thus, we raise the following question.

► **Open Problem 1.6.** Fix  $\varepsilon > 0, d \geq 1$ . Does there exist a small bias generator with error  $\gg \varepsilon^{2^d}$ , such that the sum of  $d$  independent copies of the generator does not fool degree  $d$  nonclassical polynomials with error  $\varepsilon$ ?

**Organisation.** Section 2 covers preliminary definitions. In Section 3 we prove bounds on approximation of modular sums by nonclassical polynomials. In Section 4 we analyze the approximation of the majority function by nonclassical polynomials in the correlation model and the exact computation model. In Section 5 we prove the results on the weak representation of the OR function. We describe in Appendix A an improvement in the degree of classical polynomials which weakly represent the OR function.

## 2 Preliminaries

Let  $\mathbb{N} = \{1, 2, \dots\}$  denote the set of positive integers. For  $n \in \mathbb{N}$ , let  $[n] := \{1, 2, \dots, n\}$ . Let  $\mathbb{T} = \mathbb{R}/\mathbb{Z}$  denote the torus. This is an abelian group under addition. Let  $e : \mathbb{T} \rightarrow \mathbb{C}^*$  be defined by  $e(x) = \exp(2\pi ix)$ .

### 2.1 Nonclassical polynomials

Let  $\mathbb{F}_p$  be a prime finite field. Given a function  $f : \mathbb{F}_p^n \rightarrow \mathbb{T}$ , its directional derivative in direction  $h \in \mathbb{F}_p^n$  is  $D_h f : \mathbb{F}_p^n \rightarrow \mathbb{T}$ , given by

$$D_h f(x) = f(x + h) - f(x).$$

Polynomials are defined as functions which are annihilated by repeated derivatives.

► **Definition 2.1** (Nonclassical polynomials). A function  $f : \mathbb{F}_p^n \rightarrow \mathbb{T}$  is a polynomial of degree at most  $d$  if  $D_{h_1} \dots D_{h_{d+1}} f \equiv 0$  for any  $h_1, \dots, h_{d+1} \in \mathbb{F}_p^n$ . The degree of  $f$  is the minimal  $d$  for which this holds.

Classical polynomials satisfy this definition. Let  $|\cdot|$  denote the natural map from  $\mathbb{F}_p$  to  $\{0, 1, \dots, p-1\} \subseteq \mathbb{Z}$ . If  $P : \mathbb{F}_p^n \rightarrow \mathbb{F}_p$  is a (classical) polynomial of degree  $d$ , then  $f(x) = |P(x)|/p \pmod{1}$  is a nonclassical polynomial of degree  $d$ . For degrees  $d \leq p$ , it turns out that these are the only possible polynomials. However, when  $d > p$ , there are more polynomials than just these arising from the classical ones, from which the term *nonclassical polynomials* arise. A complete characterization of nonclassical polynomials was developed by Tao and Ziegler [24]. They showed that a function  $f : \mathbb{F}_p^n \rightarrow \mathbb{T}$  is a polynomial of degree  $\leq d$  if and only if it has the following form:

$$f(x_1, \dots, x_n) = \alpha + \sum_{0 \leq e_1, \dots, e_n \leq p-1, k \geq 0: \sum e_i + (p-1)k \leq d} \frac{c_{e_1, \dots, e_n, k} |x_1|^{e_1} \dots |x_n|^{e_n}}{p^{k+1}} \pmod{1}.$$

Here,  $\alpha \in \mathbb{T}$  and  $c_{e_1, \dots, e_n, k} \in \{0, 1, \dots, p-1\}$  are uniquely determined. The coefficient  $\alpha$  is called the *shift* of  $f$ , and the largest  $k$  for which  $c_{e_1, \dots, e_n, k} \neq 0$  for some  $e_1, \dots, e_n$  is called the *depth* of  $f$ . Classical polynomials correspond to polynomials with 0 shift and 0 depth. In this work, we assume without loss of generality that all polynomials have 0 shift. Define  $\mathbb{U}_{p,k} := \frac{1}{p^k} \mathbb{Z}/\mathbb{Z}$  which is a subgroup of  $\mathbb{T}$ . Then, the image of polynomials of depth  $k-1$  lie in  $\mathbb{U}_{p,k}$ . We prove the following lemma which shows that nonclassical polynomials can be “translated” to classical polynomials of a somewhat higher degree, at least if we restrict our attention to boolean inputs.

► **Lemma 2.2.** *Let  $f : \mathbb{F}_p^n \rightarrow \mathbb{T}$  be a polynomial of degree  $d$  and depth  $\leq k-1$ . Let  $\varphi : \mathbb{U}_{p,k} \rightarrow \mathbb{F}_p$  be any function. Then there exists a classical polynomial  $g : \mathbb{F}_p^n \rightarrow \mathbb{F}_p$  of degree at most  $(p^k - 1)d$ , such that*

$$g(x) = \varphi(f(x)) \quad \forall x \in \{0, 1\}^n.$$

**Proof.** By the characterization of nonclassical polynomials, we have

$$f(x) = \sum_{e,j} \frac{c_{e,j} |x_1|^{e_1} \dots |x_n|^{e_n}}{p^j}$$

where the sum is over  $e = (e_1, \dots, e_n)$  with  $e_i \in \{0, \dots, p-1\}$ ,  $1 \leq j \leq k$  such that  $\sum e_i + (p-1)(j-1) \leq d$ . We only care about the evaluation of  $f$  on the boolean hypercube,



which allows for some simplifications. For any  $x \in \{0, 1\}^n$  we have  $|x_1|^{e_1} \dots |x_n|^{e_n} = \prod_{i \in I} x_i$  where  $I = \{i : e_i \neq 0\}$ . Thus, we can define an integer polynomial  $P(x) = \sum_I c'_I \prod_{i \in I} x_i$  such that

$$f(x) = \frac{P(x)}{p^k} \pmod{1} \quad \forall x \in \{0, 1\}^n,$$

where  $c'_I = \sum_{e: \{i: e_i \neq 0\} = I} \sum_j p^{k-j} c_{e,j}$ . In particular, note that  $P$  has degree at most  $d$ . We may further simplify  $P(x) = M_1(x) + \dots + M_t(x)$ , where each  $M_i$  is a monomial of the form  $\prod_{i \in I} x_i$ , and monomials may be repeated (indeed, the monomial  $\prod_{i \in I} x_i$  is repeated  $c'_I$  times). Hence

$$f(x) = \frac{M_1(x) + \dots + M_t(x)}{p^k} \pmod{1} \quad \forall x \in \{0, 1\}^n.$$

We care about the first  $k$  digits in base  $p$  of  $P(x) = \sum M_i(x)$ . These can be captured via the symmetric polynomials, using the fact that  $M_i(x) \in \{0, 1\}$  for all  $x \in \{0, 1\}^n$ .

The  $\ell$ -th symmetric polynomial in  $z = (z_1, \dots, z_t)$ , for  $1 \leq \ell \leq t$ , is a classical polynomial of degree  $\ell$  defined as

$$S_\ell(z) = \sum_{S \subset [t], |S| = \ell} \prod_{i \in S} z_i.$$

When  $z \in \{0, 1\}^t$ , it follows by Lucas theorem [18] that the  $i$ -th digit of  $z_1 + \dots + z_t$  in base  $p$  is given by  $S_{p^i}(z) \pmod{p}$ .

So, define a polynomial  $Q : \mathbb{F}_p^k \rightarrow \mathbb{F}_p$  such that  $Q(a_0, \dots, a_{k-1}) = \varphi(\sum a_i p^i / p^k)$  for all  $a_0, \dots, a_{k-1} \in \{0, \dots, p-1\}$ , and polynomials  $R_i : \mathbb{F}_p^n \rightarrow \mathbb{F}_p$  for  $i = 0, \dots, k-1$  by  $R_i(x) = S_{p^i}(M_1(x), \dots, M_t(x))$ . Note that  $\deg(R_i) \leq p^i d$ . Define  $g(x) = Q(R_0(x), \dots, R_{k-1}(x))$ . Then we have that

$$g(x) = \varphi(f(x)) \quad \forall x \in \{0, 1\}^n.$$

To conclude, we need to bound the degree of  $g$ . As monomials in  $Q$  raise each variable to degree at most  $p-1$ , we have  $\deg(g) \leq (p-1) \sum \deg(R_i) \leq (p^k - 1)d$ .  $\blacktriangleleft$

## 2.2 Gowers uniformity norms

Let  $F : \mathbb{F}^n \rightarrow \mathbb{C}$ . The (multiplicative) derivative of  $F$  in direction  $h \in \mathbb{F}^n$  is given by  $(\Delta_h F)(x) = F(x+h)\overline{F(x)}$ . One can verify that if  $f : \mathbb{F}^n \rightarrow \mathbb{T}$  and  $F = e(f)$  then  $\Delta_h F = e(D_h f)$ . The  $d$ -th Gowers uniformity norm  $\|\cdot\|_{U^d}$  is defined as

$$\|F\|_{U^d} := (\mathbb{E}_{h_1, \dots, h_d, x \in \mathbb{F}^n} [\Delta_{h_1} \dots \Delta_{h_d} F(x)])^{1/2^d}.$$

Observe that  $\|F\|_{U^1} = |\mathbb{E}_x[F(x)]|$ , which is a semi-norm. For  $d \geq 2$ , the Gowers uniformity norm turns out to indeed be a norm (but we will not need that). The following lists the properties of the Gowers uniformity norm that we would need. For a proof and further details, see [15].

- Let  $f : \mathbb{F}^n \rightarrow \mathbb{T}$  and  $F = e(f)$ . Then  $0 \leq \|F\|_{U^d} \leq 1$ , where  $\|F\|_{U^d} = 1$  if and only if  $f$  is a polynomial of degree  $\leq d-1$ .
- If  $f : \mathbb{F}^n \rightarrow \mathbb{T}$  is a polynomial of degree  $\leq d-1$  then  $\|Fe(f)\|_{U^d} = \|F\|_{U^d}$  for any  $F : \mathbb{F}^n \rightarrow \mathbb{C}$ .
- If  $F(x_1, \dots, x_n) = F_1(x_1) \dots F_n(x_n)$  then  $\|F\|_{U^d} = \|F_1\|_{U^d} \dots \|F_n\|_{U^d}$ .
- (Gowers-Cauchy-Schwarz) For any  $F : \mathbb{F}^n \rightarrow \mathbb{C}$  and any  $d \geq 1$ ,

$$0 \leq \|F\|_{U^1} \leq \|F\|_{U^2} \leq \dots \leq \|F\|_{U^d}.$$



### 3 Approximating modular sums by polynomials

Viola and Wigderson [26] proved that low-degree polynomials over  $\mathbb{F}_2$  cannot correlate to the sum modulo  $m$ , as long as  $m$  is odd. Their proof technique is based on the Gowers uniformity norm. As such, it extends naturally to nonclassical polynomials. We capture that by the following theorem. In the following, let  $\omega_m = \exp(2\pi i/m)$  be a primitive  $m$ -th root of unity.

► **Theorem 3.1** (Extension of [26] to nonclassical polynomials). *Let  $f : \mathbb{F}_2^n \rightarrow \mathbb{T}$  be a polynomial of degree  $< d$ . Let  $m \in \mathbb{N}$  be odd. Then for any  $a \in \{1, \dots, m-1\}$ ,*

$$\mathbb{E}_{x \in \{0,1\}^n} \left[ e(f(x)) \cdot \omega_m^{a(x_1 + \dots + x_n)} \right] \leq \exp(-cn/4^d)$$

where  $c = c_m > 0$ .

**Proof.** Let  $F(x) = e(f(x)) \cdot \omega_m^{a(x_1 + \dots + x_n)}$ . By the properties of the Gowers uniformity norm,

$$|\mathbb{E}_x[F(x)]| \leq \|F\|_{U^d} = \|\omega_m^{a(x_1 + \dots + x_n)}\|_{U^d} = \prod_{i=1}^n \|\omega_m^{ax_i}\|_{U^d} = \|e(g)\|_{U^d}^n,$$

where  $g : \mathbb{F}_2 \rightarrow \mathbb{T}$  is given by  $g(0) = 0, g(1) = a/m$ . A routine calculation shows that

$$D_{h_1} \dots D_{h_d} g(x) = \begin{cases} a'/m & \text{if } h_1 = \dots = h_d = 1, x = 0 \\ -a'/m & \text{if } h_1 = \dots = h_d = 1, x = 1 \\ 0 & \text{otherwise} \end{cases}$$

where  $a' = a2^{d-1}$  is nonzero modulo  $m$ . Hence  $\|e(g)\|_{U^d}^{2^d} = (1 - 2^{-d}) + 2^{-d} \cos(2\pi a'/m) \leq 1 - 2^{-d} \cdot \Omega(1/m^2)$  and

$$|\mathbb{E}[F]| \leq (1 - 2^{-d} \cdot \Omega(1/m^2))^{n/2^d} \leq \exp(-cn/4^d)$$

where  $c = \Omega(1/m^2)$ . ◀

This proof technique gives trivial bounds for  $d \gg \log n$ . Here, we show that this is for a good reason, as there are nonclassical polynomials of degree  $O(\log n)$  which well approximate the sum modulo  $m$ .

► **Theorem 3.2.** *Let  $m \in \mathbb{N}$  be odd and fix  $a \in \{1, \dots, m-1\}$ . For any  $\varepsilon > 0$  there exists a polynomial  $f : \mathbb{F}_2^n \rightarrow \mathbb{T}$  of degree  $\log(\frac{n+m}{\varepsilon}) + O(1)$  such that*

$$\mathbb{E}_{x \in \{0,1\}^n} \left[ e(f(x)) \cdot \omega_m^{a(x_1 + \dots + x_n)} \right] = 1 + u$$

where  $|u| \leq \varepsilon$ .

**Proof.** Let  $k \geq 1$  to be specified later. Let  $r \in \{0, \dots, m-1\}$  be such that  $r \equiv a2^k \pmod{m}$  and let  $A = \frac{r-a2^k}{m} \in \mathbb{Z}$ . Define  $f : \mathbb{F}_2^n \rightarrow \mathbb{T}$  as

$$f(x) = \frac{A(|x_1| + \dots + |x_n|)}{2^k} \pmod{1}.$$

Note that  $f$  is a polynomial of degree  $\leq k$ . For  $x \in \{0,1\}^n$ , if  $x_1 + \dots + x_n = pm + q$  where  $q \in \{0, \dots, m-1\}$ , then

$$f(x) \equiv \frac{A(pm + q)}{2^k} \equiv \frac{rp + \frac{rq}{m}}{2^k} - \frac{aq}{m} = -\frac{aq}{m} + \theta_x \pmod{1},$$

where  $0 \leq \theta_x \leq (n+m)/2^k$ . We choose  $k \geq \log(\frac{n+m}{\varepsilon}) + c$  for some absolute constant  $c > 0$  so that  $|e(\theta_x) - 1| \leq \varepsilon$  for all  $x$ . Hence

$$\left| \mathbb{E} \left[ e(f(x)) \cdot \omega_m^{a(x_1 + \dots + x_n)} \right] - 1 \right| = |\mathbb{E}[e(\theta_x) - 1]| \leq \mathbb{E}[|e(\theta_x) - 1|] \leq \varepsilon.$$

◀

#### 4 Approximating majority by nonclassical polynomials

The majority function  $\text{MAJ} : \mathbb{F}_2^n \rightarrow \mathbb{F}_2$  is defined as

$$\text{MAJ}(x) = \begin{cases} 0 & \text{if } \sum_{i=1}^n |x_i| \leq n/2 \\ 1 & \text{otherwise} \end{cases}$$

We first show that it correlates well with a nonclassical polynomial of degree  $O(\log n)$ .

► **Theorem 4.1.** *There is a nonclassical polynomial  $f : \mathbb{F}_2^n \rightarrow \mathbb{T}$  of degree  $\log n + 1$  such that*

$$\left| \mathbb{E} \left[ (-1)^{\text{MAJ}(x)} e(f(x)) \right] \right| \geq c,$$

where  $c > 0$  is an absolute constant.

**Proof.** We assume that  $n$  is even for the proof. The proof is similar for odd  $n$ . Let  $A = \lfloor a\sqrt{n} \rfloor$  for  $a > 0$  to be specified later. Let  $k$  be the smallest integer such that  $2^k \geq n$ . Set

$$f(x) = \frac{A(\sum_{i=1}^n |x_i| - n/2)}{2^k}.$$

Note that  $\deg(f) \leq \log n + 1$ . Now,

$$\begin{aligned} & \mathbb{E} \left[ (-1)^{\text{MAJ}(x)} e(f(x)) \right] \\ &= 2^{-n} \sum_{i=0}^{n/2} \binom{n}{i} e(A(i - n/2)/2^k) - 2^{-n} \sum_{i=n/2+1}^n \binom{n}{i} e(A(i - n/2)/2^k) \\ &= 2^{-n} \sum_{j=1}^{n/2} \binom{n}{n/2-j} e(-Aj/2^k) - 2^{-n} \sum_{j=1}^{n/2} \binom{n}{n/2-j} e(Aj/2^k) + 2^{-n} \binom{n}{n/2} \\ &= -2i \cdot 2^{-n} \sum_{j=1}^{n/2} \binom{n}{n/2-j} \sin(2\pi Aj/2^k) + 2^{-n} \binom{n}{n/2}, \end{aligned}$$

where in the last equation  $i = \sqrt{-1}$ . Let  $C = 2^{-n} \sum_{j=1}^{n/2} \binom{n}{n/2-j} \sin(2\pi Aj/2^k)$ , so that  $|\mathbb{E} [(-1)^{\text{MAJ}(x)} e(f(x))]| \geq 2C$ . We will show that  $C \geq \Omega(1)$ . Let  $b > 0$  be a constant to be specified later. We bound

$$C \geq 2^{-n} \sum_{j=1}^{b\sqrt{n}} \binom{n}{n/2-j} \sin(2\pi Aj/2^k) - \exp(-2b^2),$$

where the error term follows from the Chernoff bound. We set  $a = 1/8b$ . For all  $1 \leq j \leq b\sqrt{n}$  we have  $2\pi Aj/2^k \leq \pi/4$ . Applying the estimate  $\sin(x) \geq x/2$  which holds for all  $0 \leq x \leq \pi/4$ , we obtain that

$$C \geq \frac{\pi}{32b\sqrt{n}} \cdot 2^{-n} \sum_{j=1}^{b\sqrt{n}} \binom{n}{n/2-j} j - \exp(-2b^2).$$

Now, if  $b$  is a large enough constant, standard bounds on the binomial coefficients give that

$$2^{-n} \sum_{j=1}^{b\sqrt{n}} \binom{n}{n/2-j} j = \Omega(\sqrt{n}).$$

Hence, we obtain that

$$C \geq \Omega(1/b) - \exp(-2b^2).$$

If  $b$  is chosen a large enough constant, this shows that  $C \geq \Omega(1)$  as claimed. ◀

We next show that the Razborov-Smolensky technique generalizes to nonclassical polynomials when we require the polynomial to exactly compute MAJ. Recall that we identify  $\mathbb{F}_2$  with  $\{0, 1/2\} \subset \mathbb{T}$  and consider  $\text{MAJ} : \mathbb{F}_2^n \rightarrow \{0, 1/2\}$ .

► **Theorem 4.2.** *Let  $f : \mathbb{F}_2^n \rightarrow \mathbb{T}$  be a nonclassical polynomial of degree  $d$  and depth  $< k$ . Then,*

$$\Pr_{x \in \{0,1\}^n} [f(x) = \text{MAJ}(x)] \leq \frac{1}{2} + O\left(\frac{2^k d}{\sqrt{n}}\right).$$

**Proof.** Let  $\varphi : \mathbb{U}_{2,k} \rightarrow \mathbb{F}_2$  be defined as  $\varphi(0) = 0$ ,  $\varphi(1/2) = 1$  and choose arbitrarily  $\varphi(x)$  for  $x \in \mathbb{U}_{2,k} \setminus \{0, 1/2\}$ . Applying Lemma 2.2, there exists a classical polynomial  $g : \mathbb{F}_2^n \rightarrow \mathbb{F}_2$  such that  $g(x) = \varphi(f(x))$  for all  $x \in \mathbb{F}_2^n$ , where  $\deg(g) \leq (2^k - 1)d$ . In particular,

$$\Pr_{x \in \mathbb{F}_2^n} [g(x) = \text{MAJ}(x)] \geq \Pr_{x \in \mathbb{F}_2^n} [f(x) = \text{MAJ}(x)].$$

Hence, we can apply the Razborov-Smolensky bound [22, 23] to  $g$  and conclude that

$$\Pr[f(x) = \text{MAJ}(x)] \leq \frac{1}{2} + O\left(\frac{\deg(g)}{\sqrt{n}}\right).$$

◀

## 5 Weak representation of the OR function

A set of classical polynomials  $f_i : \mathbb{F}_{p_i}^n \rightarrow \mathbb{F}_{p_i}$  is said to weakly represent the OR function if they all map  $0^n$  to zero, and for any other point in the boolean hypercube, at least one of them map it to a nonzero value. This definition extends naturally to nonclassical polynomials.

► **Definition 5.1.** Let  $p_1, \dots, p_r$  be distinct primes. The polynomials  $f_i : \mathbb{F}_{p_i}^n \rightarrow \mathbb{T}$  weakly represent the OR function if

- $f_1(0^n) = \dots = f_r(0^n) = 0$ .
- For any  $x \in \{0, 1\}^n \setminus 0^n$ , there exists some  $i$  such that  $f_i(x) \neq 0$ .

It is well known that a single classical polynomial  $f : \mathbb{F}_p^n \rightarrow \mathbb{F}_p$  which weakly represents the OR function, must have degree at least  $n/(p-1)$ . This is since  $f(x)^{p-1}$  computes the OR function on  $\{0, 1\}^n$ , and hence its multi-linearization (obtained by replacing any power  $x_i^{e_i}$ ,  $e_i \geq 1$  with  $x_i$ ) must be the unique multi-linear extension of the OR function, which has degree  $n$ .

We first show that there is a nonclassical polynomial of degree  $O(\log n)$  which weakly represents the OR function.

► **Lemma 5.2.** *There exists a polynomial  $f : \mathbb{F}_p^n \rightarrow \mathbb{T}$  of degree  $O(p \lceil \log_p n \rceil)$  which weakly represents the OR function.*

**Proof.** Let  $k \geq 1$  be minimal such that  $p^k > n$ . Define  $f(x) = \frac{|x_1| + \dots + |x_n|}{p^k}$ . This is a polynomial of degree  $1 + (p-1)(k-1)$ . Clearly  $f(0^n) = 0$  and  $f(x) \neq 0$  for any  $x \in \{0, 1\}^n \setminus 0^n$ . ◀

We show that allowing for multiple nonclassical polynomials can only improve this simple construction by a polynomial factor.

► **Theorem 5.3.** *Let  $p_1, \dots, p_r$  be distinct primes, and let  $p = \max(p_1, \dots, p_r)$ . Let  $f_i : \mathbb{F}_{p_i}^n \rightarrow \mathbb{T}$  be polynomials which weakly represent the OR function. Then at least one of the polynomials must have degree  $\Omega((\log_p n)^{1/r})$ .*

The proof is an adaptation of the result of Barrington and Tardos [3], who proved similar lower bounds for classical polynomials. We start by showing that a low degree polynomial  $f$  with  $f(0) = 0$  must have another point  $x$  with  $f(x) = 0$ .

► **Claim 5.4.** *Let  $f : \mathbb{F}_p^n \rightarrow \mathbb{T}$  be a polynomial of degree  $d$  and depth  $\leq k - 1$  such that  $f(0) = 0$ . If  $n > (p^k - 1)d$  then there exists  $x \in \{0, 1\}^n \setminus 0^n$  such that  $f(x) = 0$ .*

We note that the bound on  $n$  is fairly tight, as  $f(x) = (x_1 + \dots + x_n)/p^k \pmod{1}$  violates the conclusion of the claim whenever  $n < p^k$ .

**Proof.** Let  $\varphi : \mathbb{U}_{p,k} \rightarrow \mathbb{F}_p$  be given by  $\varphi(0) = 0$ ,  $\varphi(a) = 1$  for all  $a \neq 0$ . Applying Lemma 2.2, there exists a classical polynomial  $g : \mathbb{F}_p^n \rightarrow \mathbb{F}_p$  of degree  $\leq (p^k - 1)d$  such that  $g(x) = 0$  if  $f(x) = 0$ , and  $g(x) = 1$  if  $f(x) \neq 0$ , for all  $x \in \{0, 1\}^n$ . If  $f(0^n) = 0$  but  $f(x) \neq 0$  for all nonzero  $x \in \{0, 1\}^n$ , then  $g$  computes the OR function over  $\{0, 1\}^n$ . Hence,  $\deg(g) \geq n$ , which leads to a contradiction whenever  $n > (p^k - 1)d$ . ◀

We next extend Claim 5.4 to find a common root for a number of polynomials.

► **Claim 5.5.** *Let  $f_1, \dots, f_r : \mathbb{F}_p^n \rightarrow \mathbb{T}$  be polynomials of degree  $d$  and depth  $\leq k - 1$  such that  $f_i(0) = 0$  for all  $i \in [r]$ . If  $n > (p^k - 1)dr$  then there exists  $x \in \{0, 1\}^n \setminus 0^n$  such that  $f_i(x) = 0$  for all  $i \in [r]$ .*

**Proof.** We construct an interpolating polynomial for  $f_1, \dots, f_r$ . Following the proof of Claim 5.4, for each  $f_i$  there exists a classical polynomial  $g_i : \mathbb{F}_p^n \rightarrow \mathbb{F}_p$  satisfying the following. For any  $x \in \{0, 1\}^n$ , if  $f_i(x) = 0$  then  $g_i(x) = 0$ , and if  $f_i(x) \neq 0$  then  $g_i(x) = 1$ . Moreover,  $\deg(g_i) \leq (p^k - 1)d$ . Define  $g : \mathbb{F}_p^n \rightarrow \mathbb{F}_p$  as

$$g(x) = 1 - \prod_{i=1}^r (1 - g_i(x)).$$

Note that  $\deg(g) \leq \sum \deg(g_i) \leq (p^k - 1)dr$ . Suppose for contradiction that for every  $x \in \{0, 1\}^n \setminus 0^n$  there is an  $i \in [r]$  such that  $f_i(x) \neq 0$ . Then  $g(0) = 0$  as  $f_i(0) = 0$  for all  $i \in [r]$ , but  $g(x) = 1$  for all  $x \in \{0, 1\}^n \setminus 0^n$ . Then  $g$  computes the OR function over  $\{0, 1\}^n$ , and hence  $\deg(g) \geq n$ . This leads to a contradiction whenever  $n > (p^k - 1)dr$ . ◀

Next, we argue that the hamming ball of radius  $d$  is an interpolating set for polynomials of degree  $d$  over  $\{0, 1\}^n$ . In the following, let  $B(n, d) = \{x \in \{0, 1\}^n : \sum x_i \leq d\}$ .

► **Claim 5.6.** *Let  $f : \mathbb{F}_p^n \rightarrow \mathbb{T}$  be a polynomial of degree  $d$  such that  $f(x) = 0$  for all  $x \in B(n, d)$ . Then  $f(x) = 0$  for all  $x \in \{0, 1\}^n$ .*

**Proof.** Towards contradiction, let  $x^* \in \{0, 1\}^n$  be a point such that  $f(x^*) \neq 0$ , with a minimal hamming weight. By assumption, the hamming weight of  $x^*$  is at least  $d + 1$ . Let  $i_1, \dots, i_{d+1} \in [n]$  be distinct coordinates such that  $x_{i_1}^* = \dots = x_{i_{d+1}}^* = 1$ . Let  $e_j \in \{0, 1\}^n$  be the  $j$ -th unit vector, defined as  $(e_j)_j = 1$  and  $(e_j)_{j'} = 0$  for  $j' \neq j$ . Define vectors  $h_1, \dots, h_{d+1} \in \mathbb{F}_p^n$  by  $h_j = -e_{i_j}$ . Since  $f$  is a degree  $d$  polynomial, we have

$$D_{h_1} \dots D_{h_{d+1}} f \equiv 0.$$

Evaluating this on  $x^*$  gives

$$\sum_{I \subset \{i_1, \dots, i_{d+1}\}} (-1)^{|I|} f(x^* - \sum_{i \in I} e_i) = 0.$$

However, as we chose  $x^*$  with minimal hamming weight such that  $f(x^*) \neq 0$ , we have  $f(x^* - \sum_{i \in I} e_i) = 0$  for all nonempty  $I$ . Hence also  $f(x^*) = 0$ . ◀

Next, we prove that low degree polynomials must be zero on a large combinatorial cube. In the following, we identify subsets  $S \subset [n]$  with their indicators in  $\{0, 1\}^n$ .

► **Lemma 5.7.** *Let  $f : \mathbb{F}_p^n \rightarrow \mathbb{T}$  be a polynomial of degree  $d$  and depth  $\leq k - 1$  such that  $f(0) = 0$ . For  $\ell \geq 1$ , if  $n \geq 2dp^k \ell^{d+1}$  then there exist pairwise disjoint and nonempty sets of variables  $S_1, \dots, S_\ell \subset [n]$  such that*

$$f\left(\sum_{i=1}^{\ell} y_i S_i\right) = 0 \quad \forall y \in \{0, 1\}^\ell.$$

**Proof.** Fix  $a_1, \dots, a_\ell$  to be determined later such that  $n \geq a_1 + \dots + a_\ell$ . Let  $A_1, \dots, A_\ell \subset [n]$  be disjoint subsets of variables of size  $|A_i| = a_i$ . We will find subsets  $S_i \subset A_i$  such that  $f(\sum y_i S_i) = 0$  for all  $y \in \{0, 1\}^\ell$ . As we may set the variables outside  $A_1, \dots, A_\ell$  to zero, we assume from now on that  $n = a_1 + \dots + a_\ell$ .

First, set  $a_1 = p^k d$ . Consider the restriction of  $f$  to  $A_1$  by setting the remaining variables to zero. By Claim 5.4, there exists a nonempty set  $S_1 \subset A_1$  such that  $f(S_1) = 0$ .

Next, suppose that we already constructed  $S_1 \subset A_1, \dots, S_j \subset A_j$  for some  $1 \leq j < \ell$ , such that  $f(\sum y_i S_i) = 0$  for all  $y \in \{0, 1\}^j$ . For each  $y \in \{0, 1\}^j$ , define a polynomial  $f_y : \mathbb{F}_p^{A_{j+1}} \rightarrow \mathbb{T}$  by

$$f_y(x') = f\left(\sum_{i=1}^j y_i S_i + x'\right)$$

where  $x' \in \mathbb{F}_p^{A_{j+1}}$  denotes the variables in  $A_{j+1}$ . We will find a common nonzero root for  $f_y(x')$ .

First, consider only  $y \in B(j, d)$ . The number of such polynomials is  $r = \binom{j}{\leq d} = \sum_{i=0}^d \binom{j}{i}$ . Applying claim 5.5, if we choose  $a_{j+1} \geq drp^k$  then there exists  $S_{j+1} \subset A_{j+1}$  such that

$$f_y(S_{j+1}) = 0 \quad \forall y \in B(j, d).$$

We claim that this implies that  $f_y(S_{j+1}) = 0$  for all  $y \in \{0, 1\}^j$ . To see that, define  $g : \mathbb{F}_p^j \rightarrow \mathbb{T}$  by

$$g(y) = f\left(\sum_{i=1}^j y_i S_i + S_{j+1}\right).$$

This is a polynomial of degree  $d$ , and by Claim 5.6, if it is zero for all  $y \in B(j, d)$ , then it is zero on all  $\{0, 1\}^d$ . Hence, we have that  $f(\sum_{i=1}^{j+1} y_i S_i) = 0$  for all  $y \in \{0, 1\}^{j+1}$ .

We next estimate the parameters. We have  $\binom{j}{\leq d} \leq 2j^d$ , and hence it suffices to take  $a_{j+1} = 2dj^d p^k$ . Hence, we need  $n \geq n_0$  for

$$n_0 = \sum_{j=1}^{\ell} a_j \leq 2dp^k \sum_{j=1}^{\ell} j^d \leq 2dp^k \ell^{d+1}.$$

◀

We are now ready to prove Theorem 5.3.

**Proof of Theorem 5.3.** Let  $p_1, \dots, p_r$  be distinct primes, and let  $p = \max(p_1, \dots, p_r)$ . Let  $f_i : \mathbb{F}_{p_i}^n \rightarrow \mathbb{T}$  be polynomials of degree at most  $d$  and depth at most  $k - 1$  which weakly represent the OR function. We fix integers  $n \geq n_0 = \ell_0 \geq \ell_1 \dots \geq \ell_{r-1} \geq \ell_r = 1$  which will be specified later. Applying Lemma 5.7 to  $f_1$  with parameter  $\ell_1$ , we get that as long

as  $n$  is large enough, we can find disjoint nonempty subsets  $S_{1,1}, \dots, S_{1,\ell_1} \subset [n]$  such that  $f_1(\sum y_i S_{1,i}) = 0$  for all  $y \in \{0, 1\}^{\ell_1}$ .

Next, consider the restriction of  $f_2$  to the combinatorial cube formed by  $\{S_{1,i}\}$ . That is, define  $f'_2 : \mathbb{F}_p^{\ell_1} \rightarrow \mathbb{T}$  by  $f'_2(y) = f_2(\sum y_i S_{1,i})$ . Note that  $f'_2$  is a polynomial of degree at most  $d$  and depth at most  $k - 1$ . Applying Lemma 5.7 to  $f'_2$  with parameter  $\ell_2$ , we get that as long as  $\ell_1$  is large enough, we can find disjoint nonempty subsets  $S'_{2,1}, \dots, S'_{2,\ell_2} \subset [\ell_1]$  such that  $f'_2(\sum y_i S'_{2,i}) = 0$  for all  $y \in \{0, 1\}^{\ell_2}$ . Define  $S_{2,1}, \dots, S_{2,\ell_2} \subset [n]$  by  $S_{2,i} = \cup_{j \in S'_{2,i}} S_{1,j}$ . Then  $S_{2,1}, \dots, S_{2,\ell_2}$  are disjoint nonempty subsets of  $[n]$ , such that

$$f_1 \left( \sum_{i=1}^{\ell_2} y_i S_{2,i} \right) = f_2 \left( \sum_{i=1}^{\ell_2} y_i S_{2,i} \right) = 0 \quad \forall y \in \{0, 1\}^{\ell_2}.$$

Continuing in this fashion, we ultimately find disjoint nonempty subsets  $S_{r,1}, \dots, S_{r,\ell_r} \subset [n]$  such that

$$f_1 \left( \sum_{i=1}^{\ell_r} y_i S_{r,i} \right) = \dots = f_r \left( \sum_{i=1}^{\ell_r} y_i S_{r,i} \right) = 0 \quad \forall y \in \{0, 1\}^{\ell_r}.$$

In particular,  $f_1, \dots, f_r$  cannot weakly represent the OR function. This argument requires that for each  $0 \leq i \leq r - 1$ ,  $\ell_i \geq 2dp^k \ell_{i+1}^{d+1}$ , which can be satisfied if

$$n \geq n_0 = (2dp^k)^{(d+1)^{r-1}}.$$

Now,  $k \leq d/(p-1) + 1$  and hence  $p^k \leq p^{d/(p-1)+1} \leq 2^d p$ . As we can trivially bound  $2d \leq 2^d$  we obtain the simplified bound

$$n_0 \leq 2^{d(d+1)^r \cdot \log p}.$$

Thus, if  $f_1, \dots, f_r$  weakly represent the OR function, at least one of the must have degree  $d \geq \Omega((\log_p n)^{1/r})$ .  $\blacktriangleleft$

**Acknowledgements.** We thank Parikshit Gopalan for fruitful discussions that led to the result on the classical OR representation in Appendix A. The first author would also like to thank his advisor, David Zuckerman, for his guidance and encouragement.

---

## References

- 1 Noga Alon, Oded Goldreich, Johan Hastad, and René Peralta. Simple constructions of almost  $k$ -wise independent random variables. *Random Structures and Algorithms*, 3(3):289–304, 1992.
- 2 Boaz Barak, Anup Rao, Ronen Shaltiel, and Avi Wigderson. 2source dispersers for sub-polynomial entropy and ramsey graphs beating the frankl-wilson construction. In *Proceedings of the Thirty-eighth Annual ACM Symposium on Theory of Computing*, STOC'06, pages 671–680, New York, NY, USA, 2006. ACM.
- 3 D.A. Barrington and G. Tardos. A lower bound on the mod 6 degree of the or function. *computational complexity*, 7(2):99–108, 1998.
- 4 David A. Barrington. Some problems involving Razborov-Smolensky polynomials. In *Boolean function complexity (Durham, 1990)*, volume 169 of *London Math. Soc. Lecture Note Ser.*, pages 109–128. Cambridge Univ. Press, Cambridge, 1992.
- 5 David A. Mix Barrington, Richard Beigel, and Steven Rudich. Representing Boolean functions as polynomials modulo composite numbers. *Comput. Complexity*, 4(4):367–382, 1994. Special issue on circuit complexity (Barbados, 1992).

- 6 Richard Beigel and John Gill. Counting classes: thresholds, parity, mods, and fewness. *Theoret. Comput. Sci.*, 103(1):3–23, 1992. 7th Annual Symposium on Theoretical Aspects of Computer Science (STACS 90) (Rouen, 1990).
- 7 Richard Beigel and Jun Tarui. On ACC. In *32nd Annual Symposium on Foundations of Computer Science (San Juan, PR, 1991)*, pages 783–792. IEEE Comput. Soc. Press, Los Alamitos, CA, 1991.
- 8 Abhishek Bhowmick, Zeev Dvir, and Shachar Lovett. New Bounds for Matching Vector Families. *SIAM J. Comput.*, 43(5):1654–1683, 2014.
- 9 A. Bogdanov and E. Viola. Pseudorandom bits for polynomials. In *Proc. 48<sup>th</sup> IEEE Symp. on Foundations of Computer Science (FOCS’07)*, 2007.
- 10 Zeev Dvir, Parikshit Gopalan, and Sergey Yekhanin. Matching vector codes. *SIAM J. Comput.*, 40(4):1154–1178, 2011.
- 11 Zeev Dvir and Guangda Hu. Matching-vector families and ldcs over large modulo. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques – 16th International Workshop, APPROX 2013, and 17th International Workshop, RANDOM 2013, Berkeley, CA, USA, August 21–23, 2013. Proceedings*, pages 513–526, 2013.
- 12 Klim Efremenko. 3-query locally decodable codes of subexponential length. In *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing, STOC’09*, pages 39–44, New York, NY, USA, 2009. ACM.
- 13 P. Frankl and R. M. Wilson. Intersection theorems with geometric consequences. *Combinatorica*, 1(4):357–368, 1981.
- 14 Parikshit Gopalan. Constructing ramsey graphs from boolean function representations. *Combinatorica*, 34(2):173–206, April 2014.
- 15 W.T. Gowers. A new proof of szemerédi’s theorem. *Geometric and Functional Analysis GAFA*, 11(3):465–588, 2001.
- 16 Vince Grolmusz. Superpolynomial size set-systems with restricted intersections mod 6 and explicit Ramsey graphs. *Combinatorica*, 20(1):71–85, 2000.
- 17 Shachar Lovett. Unconditional pseudorandom generators for low degree polynomials. *Theory of Computing*, 5(3):69–82, 2009.
- 18 Edouard Lucas. Théorie des fonctions numériques simplement périodiques. *American Journal of Mathematics*, 1(2):pp. 184–196, 1878.
- 19 Joseph Naor and Moni Naor. Small-bias probability spaces: Efficient constructions and applications. *SIAM journal on computing*, 22(4):838–856, 1993.
- 20 Noam Nisan. Pseudorandom bits for constant depth circuits. *Combinatorica*, 11(1):63–70, 1991.
- 21 Noam Nisan and Avi Wigderson. Hardness vs. randomness. *J. Comput. System Sci.*, 49(2):149–167, 1994.
- 22 A. A. Razborov. Lower bounds on the dimension of schemes of bounded depth in a complete basis containing the logical addition function. *Mat. Zametki*, 41(4):598–607, 623, 1987.
- 23 R. Smolensky. Algebraic methods in the theory of lower bounds for Boolean circuit complexity. In *Proc. 19<sup>th</sup> ACM Symposium on Theory of Computing (STOC’87)*, pages 77–82, 1987.
- 24 T. Tao and T. Ziegler. The inverse conjecture for the Gowers norm over finite fields in low characteristic. *ArXiv e-prints*, January 2011.
- 25 Emanuele Viola. The sum of  $d$  small-bias generators fools polynomials of degree  $d$ . *Computational Complexity*, 18(2):209–217, 2009.
- 26 Emanuele Viola and Avi Wigderson. Norms, xor lemmas, and lower bounds for polynomials and protocols. *Theory of Computing*, 4(7):137–168, 2008.
- 27 Sergey Yekhanin. Towards 3-query locally decodable codes of subexponential length. *J. ACM*, 55(1):1:1–1:16, February 2008.



## A Improved weak OR representation by classical polynomials

In this section, we construct a low degree polynomial over  $\mathbb{Z}_m$  that weakly represents the OR function. Recall that the task is to construct a polynomial  $P$  in  $\mathbb{Z}_m[x_1, \dots, x_n]$  such that  $P(0) = 0$  and  $P(x) \neq 0$  for any nonzero  $x \in \{0, 1\}^n$ . Let  $m = p_1 \cdots p_r$  for pairwise distinct primes  $p_i$ . Let  $\ell(m)$  be the largest prime divisor of  $m$ . As mentioned before, the best result is due to Barrington, Beigel and Rudich [5], who constructed a symmetric polynomial of degree  $O(\ell(m)n^{1/r})$  that weakly represents the OR function. It is also well known [5], by Lucas' theorem that for symmetric functions,  $d = \Omega(\ell(m)^{-1}n^{1/r})$ .

Our construction takes us closer to the lower bound. We construct symmetric polynomials that have modulus independent degree, that is,  $d = O(n^{1/r})$ .

► **Theorem 1.1.** *Let  $m = \prod_{i=1}^r p_i$  for pairwise distinct primes  $p_i$ . Then there exists an explicit polynomial  $P \in \mathbb{Z}_m[x_1, \dots, x_n]$  of degree at most  $2\lceil n^{1/r} \rceil$  such that  $P$  weakly represents OR modulo  $m$ .*

**Proof.** For each  $1 \leq i \leq r$ , let  $e_i$  be the smallest integer such that  $p_i^{e_i} > \lceil n^{1/r} \rceil$ . Let  $S_j$  be the  $j$ -th symmetric polynomial in  $x = (x_1, \dots, x_n)$ . Let  $\gamma_i$  be a quadratic non-residue in  $\mathbb{Z}_{p_i}$  for odd  $p_i$ . Define  $P \in \mathbb{Z}_m[x_1, \dots, x_n]$  as follows. For odd  $p_i$  define

$$P(x) = Q_i(x)^2 - \gamma_i R_i(x)^2 \pmod{p_i}$$

and for  $p_i = 2$  (if it exists) define

$$P(x) = Q_i(x)^2 + Q_i(x)R_i(x) + R_i(x)^2 \pmod{2}.$$

The polynomials  $Q_i, R_i$  are defined as

$$Q_i(x) = 1 - \prod_{j=0}^{e_i-2} (1 - S_{p_i^j}(x)^{p_i-1})$$

and

$$R_i(x) = S_{p_i^{e_i-1}}(x).$$

This uniquely defines  $P(x) \pmod{m}$ .

We next observe that  $P(x) = 0 \pmod{p_i}$  if and only if  $Q_i(x) = R_i(x) = 0 \pmod{p_i}$ . This follows from the irreducibility of  $x^2 - \gamma_i$  over  $\mathbb{Z}_{p_i}$  for odd  $p_i$  and  $x^2 + x + 1$  over  $\mathbb{Z}_2$ . In particular, as  $Q_i(0) = R_i(0) = 0$  for all  $p_i$ , we obtain that  $P(0) = 0$ . We next show that  $P(x) \neq 0$  for all  $x \in \{0, 1\}^n \setminus 0^n$ .

Let  $\text{wt}(x) := \sum_{i=1}^n |x_i|$ . If  $x \in \{0, 1\}^n \setminus 0^n$  then  $1 \leq \text{wt}(x) \leq n$ . Hence, there exists  $1 \leq i \leq r$  such that  $\text{wt}(x) \neq 0 \pmod{p_i^{e_i}}$ . To simplify notation, set  $p := p_i, e := e_i, Q := Q_i, R := R_i$  from here onwards.

Consider the  $p$ -ary expansion of  $\text{wt}(x)$ . Let  $\text{wt}(x) = \sum_{j=0}^{e-1} a_j p^j + t p^e, 0 \leq a_j \leq p-1$ . Since  $\text{wt}(x) \neq 0 \pmod{p^e}$ , we have  $a_j \neq 0$  for some  $0 \leq j \leq e-1$ . As  $x \in \{0, 1\}^n$ , we have  $S_{p^j}(x) = \binom{\text{wt}(x)}{p^j}$ . Therefore, by Lucas' theorem, we have that  $a_j = S_{p^j}(x) \pmod{p}$ .

We consider now two cases. If  $a_{e-1} \neq 0$  then  $S_{p^{e-1}}(x) \neq 0 \pmod{p}$  and hence (by definition)  $R(x) \neq 0 \pmod{p}$ . If, on the other hand,  $a_j \neq 0$  for some  $j \leq e-2$ , then  $S_{p^j}(x) \neq 0 \pmod{p}$  and hence  $Q(x) \equiv 1 \pmod{p}$ . Thus, in any case, we cannot have that both  $Q(x) = R(x) = 0 \pmod{p}$  and hence  $P(x) \neq 0 \pmod{p}$ .

To conclude, we bound the degree of  $P(x)$ . The degree of each  $Q_i(x)$  is at most  $(p_i-1) \sum_{j=0}^{e_i-2} p_i^j = p_i^{e_i-1} - 1$ . The degree of each  $R_i(x)$  is  $p_i^{e_i-1}$ . Therefore the degree of  $P(x)$



is  $\max_i 2p_i^{e_i-1}$ . This is where we improve on [5], as the upper bound that they obtain is  $p_i^{e_i}$ . Since  $e_i$  was chosen as the least integer such that  $p_i^{e_i} > \lceil n^{1/r} \rceil$ , we have that  $p_i^{e_i-1} \leq \lceil n^{1/r} \rceil$  and hence  $\deg(P) \leq 2\lceil n^{1/r} \rceil$  as claimed. ◀

# Simplified Lower Bounds on the Multiparty Communication Complexity of Disjointness

Anup Rao<sup>\*1</sup> and Amir Yehudayoff<sup>†2</sup>

- 1 Department of Computer Science and Engineering  
University of Washington, Seattle, USA  
anuprao@cs.washington.edu
- 2 Department of Mathematics  
Technion-IIT, Israel  
amir.yehudayoff@gmail.com

---

## Abstract

We show that the deterministic number-on-forehead communication complexity of set disjointness for  $k$  parties on a universe of size  $n$  is  $\Omega(n/4^k)$ . This gives the first lower bound that is linear in  $n$ , nearly matching Grolmusz's upper bound of  $O(\log^2(n) + k^2n/2^k)$ . We also simplify the proof of Sherstov's  $\Omega(\sqrt{n}/(k2^k))$  lower bound for the randomized communication complexity of set disjointness.

**1998 ACM Subject Classification** F.1.3 Complexity Measures and Classes

**Keywords and phrases** communication complexity, number-on-forehead model, set disjointness, lower bounds

**Digital Object Identifier** 10.4230/LIPIcs.CCC.2015.88

## 1 Introduction

Given a family of  $k$  sets  $\mathcal{F} = (X_1, \dots, X_k)$  over the universe  $[n]$ , the *disjointness* function is defined as

$$\text{Disjoint}(\mathcal{F}) = \begin{cases} 1 & \text{if } \bigcap_{i=1}^k X_i = \emptyset, \\ 0 & \text{otherwise.} \end{cases}$$

We study the communication complexity of computing disjointness in the number-on-forehead model [9]. We consider  $k$  parties that attempt to compute  $\text{Disjoint}(\mathcal{F})$  by exchanging messages about  $X_1, \dots, X_k$ , until one of the parties announces the value of  $\text{Disjoint}(\mathcal{F})$ . The  $i$ 'th party can see all of the inputs except for  $X_i$ , and can send messages that depend on the inputs she sees and all previous messages. All messages are visible to all parties. The communication complexity is the minimum number of bits that needs to be transmitted to compute  $\text{Disjoint}(\mathcal{F})$ . In a randomized communication protocol, the parties use shared randomness to pick a deterministic communication protocol, and then run the chosen deterministic protocol. The protocol computes  $\text{Disjoint}(\mathcal{F})$  correctly if it outputs  $\text{Disjoint}(\mathcal{F})$  with probability at least  $2/3$ , for every family  $\mathcal{F}$ . For formal definitions of multiparty communication complexity and its significance, we refer the reader to [23].

---

\* Supported by an Alfred P. Sloan Fellowship, the National Science Foundation under agreement CCF-1016565, an NSF Career award, and by the Binational Science Foundation under agreement 2010089.

† Horev Fellow – supported by the Taub Foundation. Supported by the Israel Science Foundation and by the Binational Science Foundation under agreement 2010089.

Grolmusz [19] gave a beautiful deterministic protocol showing that  $\text{Disjoint}(\mathcal{F})$  can be computed deterministically with communication  $O(\log^2(n) + k^2n/2^k)$ . Chattopadhyay [11] used similar ideas to give a protocol with communication  $O(k \log(n) + n/2^k)$ . This paper is about proving lower bounds on the communication complexity.

## 1.1 Motivation and related work

Lower bounds on multiparty communication complexity are important because several computational models such as circuits, branching programs, and propositional proofs can be used to obtain efficient communication protocols. Strong enough communication complexity lower bounds for the computation of any explicit function can therefore be used to prove lower bounds on these models [3, 15, 2, 30, 40]. In particular, lower bounds on the communication complexity of disjointness have many applications (see the recent survey [13]). Such lower bounds imply lower bounds on proof systems [6], circuit lower bounds [21, 32, 28, 39], lower bounds on communication for problems related to combinatorial auctions [16, 26, 25, 17, 20, 29], and oracle separations for complexity classes [1].

Attempts to prove lower bounds for disjointness have led to many interesting ideas. When the number of parties is  $k = 2$ , Kalyanasundaram and Schnitger [22] proved that  $\Omega(n)$  communication is required in the randomized setting. Alternate proofs and tight bounds have since been obtained [31, 4, 8] using methods involving information theory. These methods have found many other applications that we do not discuss here.

When  $k$  is large, Tesson [38] and Beame, Pitassi, Segerlind and Wigderson [7] proved that the deterministic communication complexity is  $\Omega(\log(n)/k)$ . Then Sherstov [34, 35] introduced the *pattern matrix method* for proving lower bounds in the case  $k = 2$ . The method was used to separate certain circuit classes by relating their complexity to analytic properties of boolean functions, like their approximate degree. This technique was generalized to  $k > 2$  by Chattopadhyay [10], Lee and Shraibman [24], and Chattopadhyay and Ada [12]. These last two papers proved lower bounds of the type  $\Omega(n^{1/(k+1)}/2^{2^{O(k)}})$  on the randomized communication complexity. Beame and Huynh-Ngoc [5] extended these methods further to prove that the randomized communication complexity is at least  $2^{\Omega(\sqrt{\log(n)/k})}2^{-k}$ . Finally, Sherstov [36, 37] proved the best known lower bounds prior to our work, showing that the randomized communication complexity is at least  $\Omega(\sqrt{n}/(k2^k))$ . In fact, Sherstov proved lower bounds for a broader class of functions, as we discuss below.

These results use powerful techniques such as Fourier analysis, Gowers norms, directional derivatives, and bounds on the approximate degree. The last two works of Sherstov are the main inspiration for our work.

## 1.2 Results

In what follows,  $k$  is the number of players in the number-on-forehead model, and  $n$  is the size of the universe. For an integer  $m$ , we denote by  $[m]$  the set  $\{1, 2, \dots, m\}$ , and for two real numbers  $a$  and  $b$ , we denote by  $[a, b]$  the interval  $\{x \in \mathbb{R} : a \leq x \leq b\}$ .

Our work follows the ideas in the recent papers of Sherstov [36, 37]. We prove a linear lower bound on the deterministic multiparty communication complexity of disjointness:

► **Theorem 1.1.** *The deterministic communication complexity of disjointness is  $\Omega(\frac{n}{4^k})$ .*

Given our interpretation of Sherstov's work in [36], the proof of Theorem 1.1 is short. We also simplify the proof of the randomized lower bound from [37]:

► **Theorem 1.2** ([37]). *The randomized communication complexity of disjointness is  $\Omega\left(\frac{\sqrt{n}}{k^{2k}}\right)$ .*

Sherstov proved lower bounds for functions of the type  $f(\text{Disjoint}(\mathcal{F}_1), \dots, \text{Disjoint}(\mathcal{F}_m))$ , where  $f$  is a multivariate function, and  $\mathcal{F}_1, \dots, \mathcal{F}_m$  are families on disjoint parts of the universe. In our proof, we focus on the case where  $f$  is symmetric. We symmetrize his proof by viewing  $f$  as a univariate function  $f : \{0, 1, \dots, m\} \rightarrow \{0, 1\}$ , rather than as a multivariate function.

The proof begins by bounding the *discrepancy* of the parity of several independent instances of disjointness. Here we use two different bounds that Sherstov proved [36, 37], as black boxes. The first bound, stated as Theorem 2.1 in this paper, is used for the deterministic case, and the second, stated as Theorem 2.2, is used in the randomized case.

The rest of Sherstov's proof of Theorem 1.2 is a method to control the error in an approximation of the function  $f$ , using the bounds on the discrepancy. In the symmetrized proof, this corresponds to a bound on the error in an approximation of the Kronecker delta function (i.e. the univariate function  $f$  that is the indicator of  $m$ ). We bound the error via the following theorem, which shows that every polynomial that is not correlated with any parity has a low-degree approximation:

► **Theorem 1.3.** *Let  $m$  be a power of 2. For  $j \in [m]$ , let  $J$  denote the smallest power of 2 such that  $J \geq j$ . Let  $Y_1, \dots, Y_m \in \{0, 1\}$  be distributed uniformly and independently. Suppose  $f$  is a real univariate polynomial of degree at most  $m$ , and  $\delta \geq 0$  is such that for every  $j \geq d > 0$ ,*

$$\left| \mathbb{E} [f((Y_1 + \dots + Y_j)m/J) \cdot (-1)^{Y_1 + \dots + Y_j}] \right| \leq 2^{-12J} \delta. \quad (1)$$

*Then there exists a polynomial  $g$  of degree at most  $d - 1$  such that  $|g(x) - f(x)| \leq \delta$  for all  $x \in [0, m]$ .*

To prove Theorem 1.3, we define a useful basis  $b_0(x), \dots, b_m(x)$  for the space of polynomials, where each  $b_i$  is of degree  $i$ . Given this basis, the polynomial  $g$  is just the projection of  $f$  to the space spanned by  $b_0, \dots, b_{d-1}$ . This basis may be of independent interest. For the analogous part of the proof, Sherstov finds a low-degree approximation of  $f$  using a different basis. We prove Theorem 1.3 in Section 3.

Finally, we state a corollary that may be useful in other applications.

► **Corollary 1.4.** *There exists a function  $\ell(k) \leq O(k^2 4^k)$  with the following property. Suppose each family  $\mathcal{F}_i$ ,  $i \in [m]$ , is supported on a disjoint universe of size  $\ell$ . Let  $f : \{0, 1, \dots, m\} \rightarrow \{0, 1\}$  be an arbitrary function. If the randomized communication complexity of  $f(\sum_{i=1}^m \text{Disjoint}(\mathcal{F}_i))$  is  $C$ , then there exists a polynomial  $g$  of degree at most  $C - 1$  such that  $|f(x) - g(x)| \leq 1/3 + 2^{-3C}$  for all  $x \in \{0, 1, \dots, m\}$ .*

The proof of the corollary follows easily from the ideas in Section 2.2.

## 2 The lower bounds

Without loss of generality, we assume that  $n = m\ell$ , where  $m$  is a power of 2 and  $\ell$  is a function of  $k$  to be determined. Any family  $\mathcal{F} = (X_1, \dots, X_k)$  can be described using the  $m$  families  $\mathcal{F}_1, \dots, \mathcal{F}_m$ , each over a universe of size  $\ell$ , defined as

$$\mathcal{F}_i = (X_1 \cap [(i-1)\ell + 1, i\ell], \dots, X_k \cap [(i-1)\ell + 1, i\ell]). \quad (2)$$

Distribution $\mu$ on $\mathcal{F}_1 \subseteq 2^{[\ell]}$
<p>Let <math>S_1, \dots, S_{k-1} \subseteq [\ell]</math> be uniformly random sets conditioned on <math> S_1 \cap S_2 \cap \dots \cap S_{k-1}  = 1</math>. Let <math>S_k \subseteq [\ell]</math> be uniform and independent. Set</p> $\mathcal{F}_1 = (S_1, \dots, S_{k-1}, S_k).$

■ **Figure 1** The distribution  $\mu$ .

Moreover,

$$\text{Disjoint}(\mathcal{F}) = \begin{cases} 1 & \text{if } \sum_{i=1}^m \text{Disjoint}(\mathcal{F}_i) = m, \\ 0 & \text{otherwise.} \end{cases}$$

In order to prove Theorems 1.1 and 1.2, we consider distributions on families  $\mathcal{F}$ , where each  $\mathcal{F}_i$  is independent and identically distributed. Sherstov shows that there are distributions of this type under which every protocol with small communication complexity must have low correlation with  $(-1)^{\sum_{i=1}^m \text{Disjoint}(\mathcal{F}_i)}$ .

## 2.1 The lower bound for deterministic protocols

Consider the distribution  $\mu$  given in Figure 1 as a way to sample each  $\mathcal{F}_i$ . The following theorem is an easy consequence of Theorem 4.2 in [36], and the fact that every communication protocol can be expressed as a sum of cylinder intersections:

► **Theorem 2.1** ([36]). *If each family  $\mathcal{F}_i$  is sampled independently according to  $\mu$ , and  $\pi$  is a  $k$  party protocol with communication complexity  $C$ , then*

$$\left| \mathbb{E} \left[ \pi(\mathcal{F}) \cdot (-1)^{\sum_{i=1}^m \text{Disjoint}(\mathcal{F}_i)} \right] \right| \leq 2^C \cdot \left( \frac{2^{k-1} - 1}{\sqrt{\ell}} \right)^m.$$

The proof of Theorem 2.1 involves ideas analogous to [3] and some subtle reasoning about the distribution  $\mu$ . For completeness, we give a full exposition of the proof in Appendix B. Given Theorem 2.1, Theorem 1.1 easily follows:

**Proof of Theorem 1.1.** Let  $\pi$  be a deterministic protocol that computes  $\text{Disjoint}(\mathcal{F})$  with communication complexity  $C$ . When  $\text{Disjoint}(\mathcal{F}) = 1$ , we have  $(-1)^{\sum_{i=1}^m \text{Disjoint}(\mathcal{F}_i)} = (-1)^m$  and  $\pi(\mathcal{F}) = 1$ . On the other hand, when  $\text{Disjoint}(\mathcal{F}) = 0$ , we have  $\pi(\mathcal{F}) = 0$ . In addition,  $\Pr[\text{Disjoint}(\mathcal{F}_i) = 1] = 1/2$  for all  $i \in [m]$ , which implies  $\Pr[\text{Disjoint}(\mathcal{F}) = 1] = 2^{-m}$ . Thus,

$$\left| \mathbb{E} \left[ \pi(\mathcal{F}) \cdot (-1)^{\sum_{i=1}^m \text{Disjoint}(\mathcal{F}_i)} \right] \right| = 2^{-m}. \tag{3}$$

Now set  $\ell = 16(2^{k-1} - 1)^2$ . Theorem 2.1 and (3) imply that

$$2^C \cdot ((2^{k-1} - 1)/\sqrt{\ell})^m \geq 2^{-m} \Rightarrow C \geq m = \Omega\left(\frac{n}{4^k}\right).$$

◀

Distribution $\gamma$ on $\mathcal{F}_1 \subseteq 2^{[\ell]}$
<p>We define a distribution on <math>k \times \ell</math> boolean matrices. Given a matrix <math>M</math> sampled from this distribution, define the family <math>\mathcal{F}_1</math> by setting the <math>i</math>'th set <math>S_i = \{j \in [\ell] : M_{i,j} = 1\}</math>. Let <math>t</math> be such that <math>\ell = 2^{k-1} + t(2^k - 1)</math>. Sample <math>M</math> as follows:</p> <ol style="list-style-type: none"> <li>1. For each <math>v \in \{0, 1\}^k</math> such that <math>v \neq 1^k</math>, let <math>M</math> have <math>t</math> columns equal to <math>v</math>.</li> <li>2. Let <math>b \in \{0, 1\}</math> be uniformly random. For each <math>u \in \{0, 1\}^k</math> such that <math>\sum_{i=1}^k u_i = b \pmod 2</math>, add a column to <math>M</math> that is equal to <math>u</math>.</li> <li>3. Permute the columns of <math>M</math> using a uniformly random permutation of <math>[\ell]</math>.</li> </ol>

■ **Figure 2** The distribution  $\gamma$ .

## 2.2 The lower bound for randomized protocols

The proof of Theorem 1.1 does not give anything meaningful in the randomized setting, since it may be the case that a randomized protocol has no correlation with  $(-1)^{\sum_{i=1}^m \text{Disjoint}(\mathcal{F}_i)}$ . To prove lower bounds on the randomized communication, following Sherstov, we use a more complicated distribution on inputs, as well as approximation theory.

For the rest of this section, we work with the distribution  $\gamma$  described in Figure 2. Note that under this distribution,  $\Pr[\text{Disjoint}(\mathcal{F}_1) = 1] = 1/2$ . A crucial feature of this distribution is the following symmetric structure: if  $\rho : [\ell] \rightarrow [\ell]$  is a uniformly random permutation independent of  $\mathcal{F}_1$ , then the families  $(\mathcal{F}_1, \rho(\mathcal{F}_1))$  have the same joint distribution as two independent samples  $(\mathcal{F}_1, \mathcal{F}'_1)$  from  $\gamma$  conditioned on  $\text{Disjoint}(\mathcal{F}_1) = \text{Disjoint}(\mathcal{F}'_1)$ . Here by  $\rho(\mathcal{F}_1)$  we mean the family obtained by permuting the underlying universe. In analogy with Theorem 2.1, Sherstov shows (Corollary 4.19 in [37]) that no protocol can be significantly correlated with  $(-1)^{\sum_{i=1}^m \text{Disjoint}(\mathcal{F}_i)}$  under the distribution  $\gamma$ :

► **Theorem 2.2** ([37]). *If each family  $\mathcal{F}_i$  is sampled independently according to  $\gamma$ , and  $\pi$  is a protocol with communication complexity  $C$ , then*

$$\left| \mathbb{E} \left[ \pi(\mathcal{F}) \cdot (-1)^{\sum_{i=1}^m \text{Disjoint}(\mathcal{F}_i)} \right] \right| \leq 2^C \cdot \left( \frac{c_0 k^2 4^k}{\ell} \right)^{m/4},$$

where  $c_0 > 0$  is a universal constant.

The proof of Theorem 2.2 is delicate, mainly due to the symmetric structure of the distribution  $\gamma$  (especially if one wishes to optimize the dependence on  $k$ ). This symmetric structure is, on the other hand, very useful, and we shall exploit it next.

Given any protocol  $\pi$  computing  $\text{Disjoint}(\mathcal{F})$ , define  $f_\pi$  as the unique degree  $m$  polynomial so that for all  $t \in \{0, 1, \dots, m\}$ ,

$$f_\pi(t) = \Pr \left[ \pi(\mathcal{F}) = 1 \mid \sum_{i=1}^m \text{Disjoint}(\mathcal{F}_i) = t \right]. \quad (4)$$

Since the protocol computes  $\text{Disjoint}(\mathcal{F})$  with probability at least  $2/3$ , we have that  $|f_\pi(t)| \leq 1/3$ , for  $t = 0, 1, \dots, m-1$ , and  $|1 - f_\pi(m)| \leq 1/3$ . The following well known theorem [18, 33, 27] shows that any such function must have degree  $\sqrt{m}/3$ :

► **Theorem 2.3** ([18, 33, 27]). *Let  $\epsilon \in (0, 1/2)$ . If  $f : [0, m] \rightarrow \mathbb{R}$  is a polynomial such that  $|f(t)| \leq \epsilon$  for  $t = 0, 1, \dots, m-1$ , and  $|1 - f(m)| \leq \epsilon$ , then the degree of  $f$  is at least  $\sqrt{m(1 - 2\epsilon)}/3$ .*

<b>Protocol</b> $\tau_{\pi,j}(\mathcal{F}_1, \dots, \mathcal{F}_j)$
<ol style="list-style-type: none"> <li>1. Let <math>J</math> denote the smallest power of 2 such that <math>J \geq j</math>. Note that <math>m/J</math> is an integer, since <math>m</math> is assumed to be a power of 2.</li> <li>2. Using public randomness, sample <math>J - j</math> families <math>\mathcal{F}_{j+1}, \mathcal{F}_{j+2}, \dots, \mathcal{F}_J</math> according to <math>\gamma</math>, conditioned on the event that <math>\text{Disjoint}(\mathcal{F}_{j+1}) = \text{Disjoint}(\mathcal{F}_{j+2}) = \dots = \text{Disjoint}(\mathcal{F}_J) = 0</math>.</li> <li>3. Let <math>\mathcal{G} = (\mathcal{F}_1, \dots, \mathcal{F}_1, \mathcal{F}_2, \dots, \mathcal{F}_2, \dots, \mathcal{F}_J, \dots, \mathcal{F}_J)</math> be the <math>m</math> families obtained by repeating each family <math>\mathcal{F}_i</math> exactly <math>m/J</math> times.</li> <li>4. Let <math>\rho_1, \rho_2, \dots, \rho_m : [\ell] \rightarrow [\ell], \eta : [m] \rightarrow [m]</math> be independent uniformly random permutations chosen using public randomness.</li> <li>5. Output <math>\pi(\rho_1(\mathcal{G}_{\eta(1)}), \rho_2(\mathcal{G}_{\eta(2)}), \dots, \rho_m(\mathcal{G}_{\eta(m)}))</math>.</li> </ol>

■ **Figure 3** The protocol  $\tau_{\pi,j}$ .

Theorem 2.3 is proved via a clever reduction to Markov's bound on the magnitude of derivatives in bounded polynomials. We include the short proof in Appendix A. We remark that Theorem 2.3 is tight — one can use Chebyshev polynomials to give a polynomial  $f$  of degree  $O(\sqrt{m})$  satisfying the constraints. We shall prove that if the communication of  $\pi$  is much less than  $\sqrt{n}/(k2^k)$ , then Theorems 2.2 and 1.3 imply that  $f_\pi$  can be approximated by a polynomial whose degree is much less than  $\sqrt{m}$ , contradicting Theorem 2.3.

We analyze the behavior of  $\pi$  under several carefully chosen input distributions, and use the symmetric structure of the distributions together with Sherstov's correlation bounds to show that  $f_\pi$  has low correlation with parity. We then appeal to Theorem 1.3 to conclude that  $f_\pi$  has a low degree approximation. We formalize this plan by describing  $m$  protocols  $\tau_{\pi,1}, \dots, \tau_{\pi,m}$ , each simulating  $\pi$  with a different distribution on inputs.

Define the protocol  $\tau_{\pi,j}$  as in Figure 3. The protocol  $\tau_{\pi,j}$  takes  $j$  families of sets  $\mathcal{F}_1, \dots, \mathcal{F}_j$ . If each of  $\mathcal{F}_1, \dots, \mathcal{F}_j$  is in the support of  $\gamma$ , then the protocol  $\tau_{\pi,j}$ , using shared public randomness and no communication, generates  $m$  families  $\mathcal{H}_1, \dots, \mathcal{H}_m$  that are independently distributed according to  $\gamma$ , conditioned on

$$\sum_{i=1}^m \text{Disjoint}(\mathcal{H}_i) = (m/J) \sum_{i=1}^j \text{Disjoint}(\mathcal{F}_i).$$

This distribution of  $\mathcal{H}_1, \dots, \mathcal{H}_m$  is as stated due to the symmetric structure of  $\gamma$ , which is discussed in the second paragraph of this section. Finally,  $\tau_{\pi,j}$  simulates  $\pi$  on  $\mathcal{H}_1, \dots, \mathcal{H}_m$ .

The key properties of  $\tau_{\pi,j}$  are summarized in the following lemma.

► **Lemma 2.4.** *Let  $j \leq m$ .*

1. *The communication complexity of  $\tau_{\pi,j}$  equals that of  $\pi$ .*
2. *Let  $\mathcal{F}_1, \dots, \mathcal{F}_j$  be fixed families of sets, each in the support of  $\gamma$ . Then,*

$$\Pr[\tau_{\pi,j}(\mathcal{F}_1, \dots, \mathcal{F}_j) = 1] = f_\pi \left( \left( \sum_{i=1}^j \text{Disjoint}(\mathcal{F}_i) \right) m/J \right).$$

Lemma 2.4 and Theorem 2.2 together imply that the correlation of  $f_\pi$  with parity is small:

► **Lemma 2.5.** *Let  $J$  be the smallest power of 2 so that  $J \geq j$ . If the communication complexity of  $\pi$  is  $C$ , and  $Y_1, \dots, Y_j \in \{0, 1\}$  are uniformly random and independent, then*

$$\mathbb{E} [f_\pi((Y_1 + \dots + Y_j)m/J) \cdot (-1)^{Y_1 + \dots + Y_j}] \leq 2^C \cdot \left( \frac{c_0 k^2 4^k}{\ell} \right)^{j/4},$$

where  $c_0 > 0$  is a universal constant.

**Proof.** If  $\mathcal{F}_1$  is distributed according to  $\gamma$ , then  $\text{Disjoint}(\mathcal{F}_1)$  is a uniformly random bit. Thus,

$$\begin{aligned} & \mathbb{E} [f_\pi((Y_1 + \dots + Y_j)m/J) \cdot (-1)^{Y_1 + \dots + Y_j}] \\ &= \mathbb{E} \left[ f_\pi \left( \left( \sum_{i=1}^j \text{Disjoint}(\mathcal{F}_i) \right) m/J \right) \cdot (-1)^{\sum_{i=1}^j \text{Disjoint}(\mathcal{F}_i)} \right] \end{aligned}$$

Using Lemma 2.4,

$$= \mathbb{E} \left[ \tau_{\pi, j}(\mathcal{F}_1, \dots, \mathcal{F}_j) \cdot (-1)^{\sum_{i=1}^j \text{Disjoint}(\mathcal{F}_i)} \right] \leq 2^C \cdot \left( \frac{c_0 k^2 4^k}{\ell} \right)^{j/4},$$

where the last inequality is by Theorem 2.2, since the communication complexity of  $\tau_{\pi, j}$  is equal to that of  $\pi$ . ◀

Given Lemma 2.5, the proof is completed as follows:

**Proof of Theorem 1.2.** We set  $\ell = 2^{16 \cdot 4} c_0 k^2 4^k$ , so that the right hand side of Lemma 2.5 is  $2^{C-16j}$ . Fix any randomized protocol  $\pi$  that computes  $\text{Disjoint}(\mathcal{F})$  on the distribution induced by  $\gamma$  with  $C$  bits of communication. Let  $f_\pi$  be as defined in (4).

By Lemma 2.5,  $f_\pi$  satisfies the hypothesis of Theorem 1.3, with  $d = C$ . Thus we conclude that there is a degree  $C-1$  polynomial  $g$  that agrees with  $f_\pi$  up to an error of  $2^{-3C}$ . Theorem 2.3 implies that

$$C \geq \sqrt{m(1 - 2(1/3 + 2^{-3C}))/3} \Rightarrow C \geq \Omega(\sqrt{n}/(k2^k)).$$

◀

### 3 Approximating functions that are not correlated with parity

Here we prove Theorem 1.3, which shows that if a polynomial has low correlation with parity, then it can be approximated by a low degree polynomial. We restate the theorem for convenience.

► **Theorem 1.3 (restated).** *Let  $m$  be a power of 2. For  $j \in [m]$ , let  $J$  denote the smallest power of 2 such that  $J \geq j$ . Let  $Y_1, \dots, Y_m \in \{0, 1\}$  be distributed uniformly and independently. Suppose  $f$  is a real univariate polynomial of degree at most  $m$ , and  $\delta \geq 0$  is such that for every  $j \geq d > 0$ ,*

$$\left| \mathbb{E} [f((Y_1 + \dots + Y_j)m/J) \cdot (-1)^{Y_1 + \dots + Y_j}] \right| \leq 2^{-12J} \delta. \quad (5)$$

*Then there exists a polynomial  $g$  of degree at most  $d-1$  such that  $|g(x) - f(x)| \leq \delta$  for all  $x \in [0, m]$ .*



In what follows, let  $Y_1, \dots, Y_m \in \{0, 1\}$  be independent and uniformly random bits, and let  $I, J$  be the smallest powers of 2 such that  $I \geq i$  and  $J \geq j$ .

To prove the theorem, we define a useful basis for the space of polynomials. Let  $b_0(x) = 1$ . For  $i > 0$ , let<sup>1</sup>

$$b_i(x) = 2^i \binom{xI/m}{i} = \frac{2^i x(x - m/I)(x - 2m/I) \dots (x - (i - 1)m/I)}{i! \cdot (m/I)^i}.$$

Since  $b_i$  is of degree  $i$ , the polynomials  $b_0, \dots, b_m$  form a basis for the space of polynomials of degree at most  $m$ . To prove Theorem 1.3, we express  $f$  in this basis and then argue that all coefficients corresponding to high degree terms are negligible. The polynomials in our basis can be bounded by the following lemma:

► **Lemma 3.1.** *For every  $i \in \{0, 1, \dots, m\}$ ,  $\max_{x \in [0, m]} |b_i(x)| \leq 8^i$ .*

**Proof.** We show that the maximum of  $b_i$  is attained when  $x = m$ , and so

$$\max_{x \in [0, m]} |b_i(x)| = |b_i(m)| = 2^i \binom{mI/m}{i} \leq 2^i \cdot 2^I \leq 8^i.$$

Note that the magnitude of  $b_i$  is symmetric around the point  $(i - 1)m/(2I)$ ,

$$|b_i(x + (i - 1)m/(2I))| = |b_i(-x + (i - 1)m/(2I))|.$$

So the maximum is attained with  $x \in [(i - 1)m/(2I), m]$ . For any such  $x$  that is not a root of  $b_i$ ,

$$\left| \frac{b_i(x + m/I)}{b_i(x)} \right| = \left| \frac{x + m/I}{x - (i - 1)m/I} \right| \geq \left| \frac{x + m/I}{x} \right| > 1,$$

proving that the maximum is attained with  $x \in [m - m/I, m]$ . For such  $x$ , every term  $(x - jm/I)$  with  $j \in \{0, 1, \dots, i - 1\}$  in  $b_i(x)$  is non-negative, and so the maximum is attained when  $x = m$ . ◀

The basis polynomials behave nicely under the random experiments from (1):

► **Lemma 3.2.** *For all  $i \in \{0, 1, 2, \dots, m\}$  and  $j \in [m]$ ,*

$$|\mathbb{E} [b_i((Y_1 + \dots + Y_j)m/J) \cdot (-1)^{Y_1 + \dots + Y_j}]| \begin{cases} = 0 & \text{if } i < j, \\ = 1 & \text{if } i = j, \\ = 0 & \text{if } j < i \leq J, \\ \leq 8^i & \text{if } J < i. \end{cases}$$

**Proof.** When  $i < j$ , the polynomial  $b_i(y_1 + \dots + y_j)$  has degree  $i$  in the variables  $y_1, \dots, y_j$ . Since every monomial must exclude one of the  $j$  variables, the contribution of each of the monomials to the expectation is 0. When  $i = j$ ,  $b_i((y_1 + \dots + y_i)m/I) = 2^i \binom{y_1 + \dots + y_i}{i}$  is non-zero only when  $y_1 = y_2 = \dots = y_i = 1$ . Thus the expectation is  $2^{-i} \cdot 2^i \binom{i}{i} = 1$  in this case. When  $j < i \leq J$ , we have  $I = J$ . Since for  $r \in [i - 1]$ ,  $b_i(rm/I) = 2^i \binom{r}{i} = 0$ , the expectation is 0. When  $i > J$ , by Lemma 3.1, the expectation is at most  $8^i$ . ◀

Theorem 1.3 now follows by straightforward induction:

<sup>1</sup> Here and below we think of  $\binom{x}{i} = \frac{x(x-1)(x-2)\dots(x-(i-1))}{i!}$  as a real polynomial in the variable  $x$ .

**Proof of Theorem 1.3.** Write  $f(x) = \sum_{j=0}^m a_j b_j(x)$ , and let  $g(x)$  be the degree  $d-1$  polynomial  $g(x) = \sum_{j=0}^{d-1} a_j b_j(x)$ . To prove the theorem, we show that  $|g(x) - f(x)| \leq \delta$  for all  $x \in [0, m]$ . Lemma 3.2 and (1) imply that for  $j = d, \dots, m$ ,

$$\begin{aligned} |a_j| - \sum_{i=J+1}^m 8^i |a_i| &\leq |\mathbb{E}[f((Y_1 + \dots + Y_j)m/J) \cdot (-1)^{Y_1 + \dots + Y_j}]| \leq 8^{-4J} \delta \\ \Rightarrow |a_j| &\leq 8^{-4J} \delta + \sum_{i=J+1}^m 8^i |a_i|. \end{aligned} \quad (6)$$

We now prove by induction that for  $j = m, m-1, \dots, d$ ,

$$\sum_{t=j}^J |a_t| \leq 8^{-3J} \delta. \quad (7)$$

When  $m/2 < j \leq m$ , (6) implies

$$\sum_{t=j}^m |a_t| \leq (m/2) 8^{-4m} \delta \leq 8^{-3m} \delta.$$

In the general case (6) implies

$$(2/J) \sum_{t=j}^J |a_t| \leq 8^{-4J} \delta + \sum_{t=J+1}^m 8^t |a_t| \leq 8^{-4J} \delta + \sum_{r=\log(J)+1}^{\log(m)} 8^{2^r} \sum_{t=1+2^{r-1}}^{2^r} |a_t|$$

Applying the induction hypothesis, we get

$$\leq 8^{-4J} \delta + \sum_{r=\log(J)+1}^{\log(m)} 8^{2^r} 8^{-3 \cdot 2^r} \delta \leq 8^{-4J} \delta + 8^{-4J} \delta \sum_{q=0}^{\infty} 8^{-q} \leq 8^{-3J} (2/J) \delta,$$

which proves the general case of (7).

Finally, for every  $x \in [0, m]$ , Lemma 3.1 and (7) imply

$$|g(x) - f(x)| \leq \sum_{j=d}^m |a_j| |b_j(x)| \leq \sum_{r=\lceil \log d \rceil}^{\log m} 8^{-3 \cdot 2^r} \delta \cdot 8^{2^r} \leq 8^{-2d} \delta \sum_{q=0}^{\infty} 8^{-2q} \leq \delta.$$

◀

**Acknowledgements.** We thank Paul Beame, Pavel Hrubeš, and Alexander Sherstov for useful discussions.

---

## References

- 1 Scott Aaronson and Avi Wigderson. Algebrization: A new barrier in complexity theory. *TOCT*, 1(1), 2009.
- 2 László Babai, Thomas P. Hayes, and Peter G. Kimmel. The cost of the missing bit: Communication complexity with help. *Combinatorica*, 21(4):455–488, 2001.
- 3 László Babai, Noam Nisan, and Mario Szegedy. Multiparty protocols, pseudorandom generators for logspace, and time-space trade-offs. *J. Comput. Syst. Sci.*, 45(2):204–232, 1992.
- 4 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.

- 5 Paul Beame and Dang-Trinh Huynh-Ngoc. Multipart communication complexity and threshold circuit size of  $AC^0$ . In *FOCS*, pages 53–62, 2009.
- 6 Paul Beame, Toniann Pitassi, and Nathan Segerlind. Lower bounds for Lovász–Schrijver systems and beyond follow from multipart communication complexity. *SIAM Journal on Computing*, 37(3):845–869, 2007.
- 7 Paul Beame, Toniann Pitassi, Nathan Segerlind, and Avi Wigderson. A strong direct product theorem for corruption and the multipart communication complexity of disjointness. *Computational Complexity*, 15(4):391–432, 2006.
- 8 Mark Braverman and Ankur Moitra. An information complexity approach to extended formulations. In *STOC*, pages 161–170. ACM, 2013.
- 9 Ashok K. Chandra, Merrick L. Furst, and Richard J. Lipton. Multi-party protocols. In *STOC*, pages 94–99, 1983.
- 10 Arkadev Chattopadhyay. Discrepancy and the power of bottom fan-in in depth-three circuits. In *FOCS*, pages 449–458. IEEE Computer Society Press, 2007.
- 11 Arkadev Chattopadhyay. *Circuits, Communication and Polynomials*. PhD thesis, University of Toronto, 2008.
- 12 Arkadev Chattopadhyay and Anil Ada. Multipart communication complexity of disjointness. *CoRR*, abs/0801.3624, 2008.
- 13 Arkadev Chattopadhyay and Toniann Pitassi. The story of set disjointness. *SIGACT News*, 41(3):59–85, 2010.
- 14 Elliot W. Cheney. *Introduction to Approximation Theory*. McGraw-Hill Book Co., New York, 1966.
- 15 Fan R. K. Chung and Prasad Tetali. Communication complexity and quasi randomness. *SIAM Journal on Discrete Mathematics*, 6(1):110–123, February 1993.
- 16 Vincent Conitzer and Tuomas Sandholm. Communication complexity as a lower bound for learning in games. In Carla E. Brodley, editor, *ICML*, volume 69 of *ACM International Conference Proceeding Series*. ACM, 2004.
- 17 Shahar Dobzinski and Noam Nisan. Limitations of VCG-based mechanisms. *Combinatorica*, 31(4):379–396, 2011.
- 18 H. Ehlich and K. Zeller. Schwankung von polynomen zwischen gitterpunkten. *Mathematische Zeitschrift*, 86:41–44, 1964.
- 19 Vince Grolmusz. The bns lower bound for multi-party protocols in nearly optimal. *Inf. Comput.*, 112(1):51–54, 1994.
- 20 Sergiu Hart and Yishay Mansour. The communication complexity of uncoupled nash equilibrium procedures. In David S. Johnson and Uriel Feige, editors, *STOC*, pages 345–353. ACM, 2007.
- 21 Johan Håstad and Mikael Goldmann. On the power of small-depth threshold circuits. *Computational Complexity*, 1:113–129, 1991.
- 22 Bala Kalyanasundaram and Georg Schnitger. The probabilistic communication complexity of set intersection. *SIAM Journal on Discrete Mathematics*, 5(4):545–557, November 1992.
- 23 Eyal Kushilevitz and Noam Nisan. *Communication complexity*. Cambridge University Press, 1997.
- 24 Troy Lee and Adi Shraibman. Disjointness is hard in the multipart number-on-the-forehead model. *Computational Complexity*, 18(2):309–336, 2009.
- 25 Noam Nisan. The communication complexity of approximate set packing and covering. *Lecture Notes in Computer Science*, 2380:868–875, 2002.
- 26 Noam Nisan and Ilya Segal. The communication requirements of efficient allocations and supporting prices. *J. Economic Theory*, 129(1):192–224, 2006.
- 27 Noam Nisan and Mario Szegedy. On the degree of Boolean functions as real polynomials. *Computational Complexity*, 4(4):301–313, 1994.

- 28 Noam Nisan and Avi Wigderson. Rounds in communication complexity revisited. *SIAM Journal on Computing*, 22(1):211–219, February 1993.
- 29 Christos H. Papadimitriou, Michael Schapira, and Yaron Singer. On the hardness of being truthful. In *FOCS*, pages 250–259. IEEE Computer Society, 2008.
- 30 Ran Raz. The BNS-chung criterion for multi-party communication complexity. *Computational Complexity*, 9(2):113–122, 2000.
- 31 Alexander A. Razborov. On the distributional complexity of disjointness. *Theor. Comput. Sci.*, 106(2):385–390, 1992.
- 32 Alexander A. Razborov and Avi Wigderson.  $n^\omega(\log n)$  lower bounds on the size of depth-3 threshold circuits with AND gates at the bottom. *Information Processing Letters*, 45(6):303–307, 1993.
- 33 Theodore J. Rivlin and Elliott W. Cheney. A comparison of uniform approximations on an interval and a finite subset thereof. *SIAM Journal on Numerical Analysis*, 3(2):311–320, June 1966.
- 34 Alexander A. Sherstov. Separating  $AC^0$  from depth-2 majority circuits. *SIAM Journal of Computing*, 38(6):2113–2129, 2009.
- 35 Alexander A. Sherstov. The pattern matrix method. *SIAM Journal of Computing*, 40(6):1969–2000, 2011.
- 36 Alexander A. Sherstov. The multiparty communication complexity of set disjointness. In *STOC*, pages 525–548, 2012.
- 37 Alexander A. Sherstov. Communication lower bounds using directional derivatives. In *STOC*, pages 921–930, 2013.
- 38 Pascal Tesson. *Computational complexity questions related to finite monoids and semi-groups*. PhD thesis, McGill University, 2003.
- 39 Emanuele Viola. Pseudorandom bits for constant-depth circuits with few arbitrary symmetric gates. *SIAM Journal on Computing*, 36(5):1387–1403, 2007.
- 40 Emanuele Viola and Avi Wigderson. Norms, XOR lemmas, and lower bounds for polynomials and protocols. *Theory of Computing*, 4(1):137–168, 2008.

## A

 Approximation theory

The proof relies on a fundamental theorem of Markov, relating the degree of a bounded polynomial to the maximum value of its derivative.

► **Theorem 1.1** (Markov’s Theorem [14]). *Let  $g : [-1, 1] \rightarrow [-1, 1]$  be computed by a polynomial of degree  $d$ . Then  $|g'(y)| \leq d^2$  for every  $y \in [-1, 1]$ .*

Markov’s theorem allows us to prove the statement about approximation that we need:

**Proof of Theorem 2.3.** Let  $d$  be the degree of  $f$  and let  $D = \max_{x \in [0, m]} |f'(x)|$ . We can bound  $f$  using  $D$  as follows. The value  $|f(j)|$  is at most  $1 + \epsilon$  for  $j \in \{0, 1, \dots, m\}$ , and so  $|f(x)| \leq 1 + \epsilon + D/2$  for  $x \in [0, m]$ . On the other hand,  $D \geq \frac{f(m) - f(m-1)}{1} \geq 1 - 2\epsilon$ .

Now consider the degree  $d$  polynomial  $g : [-1, 1] \rightarrow [-1, 1]$  given by  $g(y) = \frac{f(my/2 + m/2)}{1 + \epsilon + D/2}$ . Since  $g'(y) = \frac{(m/2)f'(my/2 + m/2)}{1 + \epsilon + D/2}$ , there is a  $y \in [-1, 1]$  such that  $|g'(y)| = \frac{Dm/2}{1 + \epsilon + D/2}$ . By Theorem 1.1,

$$d^2 \geq \frac{Dm/2}{1 + \epsilon + D/2} \geq \frac{m(1/2 - \epsilon)}{1 + \epsilon + 1/2 - \epsilon} = \frac{2m(1/2 - \epsilon)}{3},$$

so

$$d \geq \sqrt{m(1 - 2\epsilon)/3}.$$

◀

## B Bounding the discrepancy for the deterministic case

Here we give the proof of Theorem 2.1 [36]. Let  $\mathcal{F} = (T_1, \dots, T_k)$ . We shall need to analyze the discrepancy on a more general class of distributions. Let each family  $\mathcal{F}_i$  be sampled independently according to the distribution  $\mu$ , on a universe of size  $\ell_i$ . Let  $g(\mathcal{F})$  be a cylinder intersection, that is,  $g(\mathcal{F}) = \prod_{i=1}^k g_i(\mathcal{F})$  where each  $g_i$  is 0/1 valued and does not depend on  $T_i$ . We shall prove that

$$\left| \mathbb{E} \left[ g(\mathcal{F}) \cdot (-1)^{\sum_{i=1}^m \text{Disjoint}(\mathcal{F}_i)} \right] \right| \leq \prod_{i=1}^m \frac{2^{k-1} - 1}{\sqrt{\ell_i}}, \quad (8)$$

which implies Theorem 2.1, since every communication protocol with communication  $C$  can be expressed as a sum of  $2^C$  cylinder intersections. We prove (8) by induction on  $k$ .

When  $k = 2$ , convexity implies that

$$\begin{aligned} & \left| \mathbb{E} \left[ g(\mathcal{F}) \cdot (-1)^{\sum_{i=1}^m \text{Disjoint}(\mathcal{F}_i)} \right] \right|^2 \\ & \leq \mathbb{E}_{T_2} \left[ g_1(\mathcal{F}) \mathbb{E}_{T_1} \left[ g_2(\mathcal{F}) \cdot (-1)^{\sum_{i=1}^m \text{Disjoint}(\mathcal{F}_i)} \right]^2 \right] \\ & \leq \mathbb{E}_{T_2} \left[ \mathbb{E}_{T_1} \left[ g_2(\mathcal{F}) \cdot (-1)^{\sum_{i=1}^m \text{Disjoint}(\mathcal{F}_i)} \right]^2 \right] \\ & = \mathbb{E}_{T_2, T_1, T'_1} \left[ g_2(\mathcal{F}) \cdot g_2(\mathcal{F}') \cdot (-1)^{\sum_{i=1}^m \text{Disjoint}(\mathcal{F}_i) + \text{Disjoint}(\mathcal{F}'_i)} \right] \\ & \leq \mathbb{E}_{T_1, T'_1} \left[ \left| \mathbb{E}_{T_2} \left[ (-1)^{\sum_{i=1}^m \text{Disjoint}(\mathcal{F}_i) + \text{Disjoint}(\mathcal{F}'_i)} \right] \right| \right], \end{aligned}$$

where here  $\mathcal{F} = (T_1, T_2)$  and  $\mathcal{F}' = (T'_1, T_2)$ . Now for every fixing of  $T_1, T'_1$ , the inner expectation is 1 when  $T_1 = T'_1$ , and otherwise it is 0. Thus,

$$\left| \mathbb{E} \left[ g(\mathcal{F}) \cdot (-1)^{\sum_{i=1}^m \text{Disjoint}(\mathcal{F}_i)} \right] \right|^2 \leq \Pr[T_1 = T'_1] = \prod_{i=1}^m \frac{1}{\ell_i},$$

proving the base case.

When  $k > 2$ , we again use convexity to bound

$$\begin{aligned} & \left| \mathbb{E} \left[ g(\mathcal{F}) \cdot (-1)^{\sum_{i=1}^m \text{Disjoint}(\mathcal{F}_i)} \right] \right|^2 \\ & \leq \mathbb{E}_{T_2, \dots, T_k} \left[ g_1(\mathcal{F}) \mathbb{E}_{T_1} \left[ \prod_{j=2}^k g_j(\mathcal{F}) \cdot (-1)^{\sum_{i=1}^m \text{Disjoint}(\mathcal{F}_i)} \right]^2 \right] \\ & \leq \mathbb{E}_{T_1, T'_1} \left[ \left| \mathbb{E}_{T_2, \dots, T_k} \left[ \left( \prod_{j=2}^k g_j(\mathcal{F}) \cdot g_j(\mathcal{F}') \right) (-1)^{\sum_{i=1}^m \text{Disjoint}(\mathcal{F}_i) + \text{Disjoint}(\mathcal{F}'_i)} \right] \right| \right], \quad (9) \end{aligned}$$

where here  $\mathcal{F} = (T_1, T_2, \dots, T_k)$  and  $\mathcal{F}' = (T'_1, T_2, \dots, T_k)$ . Recall that the first  $k-1$  sets of  $\mathcal{F}_i$  and  $\mathcal{F}'_i$  each intersect in exactly one element. Let  $Z = (Z_1, Z_2, \dots, Z_m)$ , where  $Z_i$  is the indicator random variable for the event that these two elements are not the same in  $\mathcal{F}_i$  and  $\mathcal{F}'_i$ . Let  $Q = T_1 \setminus T'_1$ ,  $Q' = T'_1 \setminus T_1$ , and denote by  $Q_i, Q'_i$  the intersection of these sets with the  $i$ 'th part of the universe (see (2)). Let  $R$  denote all the intersections of the sets  $T_2, \dots, T_k$

with the elements that are not in  $Q, Q'$ . By convexity of the absolute value function,

$$(9) \leq \mathbb{E}_{T_1, T'_1, R, Z} \left[ \left[ \mathbb{E}_{T_2, \dots, T_k} \left[ \left( \prod_{j=2}^k g_j(\mathcal{F}) \cdot g_j(\mathcal{F}') \right) (-1)^{\sum_{i=1}^m \text{Disjoint}(\mathcal{F}_i) + \text{Disjoint}(\mathcal{F}'_i)} \right] \right] \right] \\ \leq \mathbb{E}_{Z, Q, Q'} \left[ \prod_{i: Z_i=1} \frac{(2^{k-2} - 1)^2}{\sqrt{|Q_i| \cdot |Q'_i|}} \right], \quad (10)$$

where the last inequality follows from the fact that after fixing  $T_1, T'_1, Z, R$ , the inner expectation can be bounded by the inductive hypothesis applied to the families where  $Z_i = 1$ , over the disjoint universes  $Q_i, Q'_i$ , and the cylinder intersection defined by  $\prod_{j=2}^k g_j(\mathcal{F})g_j(\mathcal{F}')$ . Apply the arithmetic-mean-geometric-mean inequality to conclude that

$$(10) \leq \mathbb{E}_{Z, Q, Q'} \left[ \prod_{i: Z_i=1} (2^{k-2} - 1)^2 \frac{1}{2} \left( \frac{1}{|Q_i|} + \frac{1}{|Q'_i|} \right) \right]. \quad (11)$$

Since (even conditioned on the value of  $Z$ ) the size of  $Q_i$  is distributed identically to the size of  $Q'_i$ , we have

$$(11) = \mathbb{E}_{Z, Q} \left[ \prod_{i: Z_i=1} \frac{(2^{k-2} - 1)^2}{|Q_i|} \right] \\ \leq \prod_{i=1}^m \left( \Pr[Z_i = 0] + \mathbb{E}_{Z_i, Q_i} \left[ \frac{Z_i(2^{k-2} - 1)^2}{|Q_i|} \right] \right), \quad (12)$$

where here we adopt the convention that  $Z_i/|Q_i|$  is 0 when  $Z_i = 0, |Q_i| = 0$ . We shall prove that for all  $i$ ,

$$\Pr[Z_i = 0] \leq \frac{2^{k-1} - 1}{\ell_i}, \quad (13)$$

and

$$\mathbb{E}_{Z_i, Q_i} \left[ \frac{Z_i}{|Q_i|} \right] \leq \frac{2(2^{k-1} - 1)}{\ell_i(2^{k-2} - 1)}. \quad (14)$$

Inequalities (13) and (14) imply that

$$(12) \leq \prod_{i=1}^m \left( \frac{2^{k-1} - 1}{\ell_i} + \frac{2(2^{k-1} - 1)(2^{k-2} - 1)^2}{\ell_i(2^{k-2} - 1)} \right) \\ = \prod_{i=1}^m \left( \frac{2^{k-1} - 1 + (2^{k-1} - 1)(2^{k-1} - 2)}{\ell_i} \right) = \prod_{i=1}^m \frac{(2^{k-1} - 1)^2}{\ell_i},$$

as required.

It only remains to prove (13) and (14). Fix  $i$  for the rest of the proof. We start with (13). Let  $S_1, \dots, S_k$  be the intersections of  $T_1, \dots, T_k$  with the  $i$ 'th part of the universe, and let  $S'_1$  be the intersection of  $T'_1$  with the  $i$ 'th part of the universe. Observe that for all  $w \neq \emptyset$ ,

$$\Pr \left[ Z_i = 0 \mid \bigcap_{j=2}^{k-1} S_j = w \right] = \frac{1}{|w|}.$$

The total number of choices for sets  $S_1, \dots, S_{k-1}$  can be counted as the number of ways to pick the common intersection point, times the number of configurations for the rest of the

universe:  $\ell_i \cdot (2^{k-1} - 1)^{\ell_i - 1}$ . Of these, the number of configurations with  $\bigcap_{j=2}^{k-1} S_j = w$  can be counted as the number of choices for the common intersection point in  $w$ , times the number of configurations for the rest of the universe:  $|w| \cdot (2^{k-1} - 2)^{\ell_i - |w|}$ . So the probability that the two intersection points are the same is

$$\begin{aligned} \Pr[Z_i = 0] &= \frac{1}{\ell_i \cdot (2^{k-1} - 1)^{\ell_i - 1}} \cdot \sum_{w \neq \emptyset} \frac{|w| \cdot (2^{k-1} - 2)^{\ell_i - |w|}}{|w|} \\ &\leq \frac{1}{\ell_i \cdot (2^{k-1} - 1)^{\ell_i - 1}} \cdot (2^{k-1} - 2 + 1)^{\ell_i} = \frac{2^{k-1} - 1}{\ell_i}, \end{aligned}$$

proving (13).

Next we prove (14). Let  $p = \frac{2^{k-2} - 1}{2(2^{k-1} - 1)}$ . Let  $V = S_1 \cap \dots \cap S_{k-1}$  be the intersection set of size 1. We claim that for every non-empty set  $q$ , and singleton  $v$ ,

$$\Pr[Z_i = 1, Q_i = q, V = v] \begin{cases} \leq p^{|q|-1} (1-p)^{\ell_i - |q|} / \ell_i & \text{when } v \subseteq q, \\ = 0 & \text{otherwise.} \end{cases} \quad (15)$$

When  $V$  is not contained in  $Q_i$ , the value of  $Z_i$  is always 0. On the other hand, when  $v \subseteq q$ ,

$$\begin{aligned} \Pr[Z_i = 1, Q_i = q, V = v] &\leq \Pr[Q_i = q, V = v] \\ &= (1/\ell_i) \cdot \Pr[Q_i = q | V = v] \\ &\leq p^{|q|-1} (1-p)^{\ell_i - |q|} / \ell_i, \end{aligned}$$

since every element  $e \notin v$  is included in  $Q_i$  with probability  $p$ , independent of all other such elements. Indeed such an element is included in  $S_1$  with probability  $\frac{2^{k-2} - 1}{2^{k-1} - 1}$ , and given that it is included in  $S_1$ , it is excluded from  $S'_1$  with probability  $1/2$ .

When  $|Q_i| = 0$ , we have that  $Z_i = 0$ , and so  $Z_i/|Q_i| = 0$  by our convention. Thus (15) gives

$$\begin{aligned} \mathbb{E}_{Z, Q_i} \left[ \frac{Z_i}{|Q_i|} \right] &\leq \sum_{v: |v|=1} \sum_{q: v \subseteq q} \frac{p^{|q|-1} (1-p)^{\ell_i - |q|} / \ell_i}{|q|} \\ &= \frac{1}{p \ell_i} \sum_{q \neq \emptyset} p^{|q|} (1-p)^{\ell_i - |q|} \\ &\leq \frac{1}{p \ell_i} \cdot (1-p+p)^{\ell_i} = \frac{2(2^{k-1} - 1)}{\ell_i(2^{k-2} - 1)}, \end{aligned}$$

proving (14).

# How to Compress Asymmetric Communication

Sivaramakrishnan Natarajan Ramamoorthy\* and Anup Rao†

University of Washington  
Seattle, WA, USA  
{sivanr, anuprao}@cs.washington.edu

---

## Abstract

We study the relationship between communication and information in 2-party communication protocols when the information is asymmetric. If  $I^A$  denotes the number of bits of information revealed by the first party,  $I^B$  denotes the information revealed by the second party, and  $C$  is the number of bits of communication in the protocol, we show that

- one can simulate the protocol using order  $I^A + \sqrt[4]{C^3 \cdot I^B} \cdot \log C + \sqrt{C \cdot I^B} \cdot \log C$  bits of communication,
- one can simulate the protocol using order  $I^A \cdot 2^{O(I^B)}$  bits of communication.

The first result gives the best known bound on the complexity of a simulation when  $I^A \gg I^B, C^{3/4}$ . The second gives the best known bound when  $I^B \ll \log C$ . In addition we show that if a function is computed by a protocol with asymmetric information complexity, then the inputs must have a large, nearly monochromatic rectangle of the right dimensions, a fact that is useful for proving lower bounds on lopsided communication problems.

**1998 ACM Subject Classification** F.1.2 Modes of Computation

**Keywords and phrases** Communication Complexity, Interactive Compression, Information Complexity

**Digital Object Identifier** 10.4230/LIPIcs.CCC.2015.102

## 1 Introduction

Can one compress the communication in 2-party communication protocols when the information revealed by the messages of the protocol is small? This has been a central question in communication complexity in recent years. Interest in the question is fueled by applications to proving lower bounds in communication complexity, which in turn yield lower bounds for streaming algorithms and data structures among other models. In this work, we obtain stronger results when one party reveals much less information than the other, a case that often arises when studying data structures.

Throughout this discussion we assume that there is a known distribution on inputs to a communication protocol. The question of compressing protocols was posed by Chakrabarti, Shi, Wirth and Yao [10] (see also [3, 1, 21]), who defined the information cost of the protocol as the mutual information between the inputs and the messages of the protocol. Barak, Braverman, Chen and Rao [4] called this measure the *external* information cost of a protocol, and proved that if the protocol  $\pi$  has external information cost  $I_\pi^{\text{ext}}$  and communication  $C_\pi$ , then one can simulate the protocol with  $O(I_\pi^{\text{ext}} \log C_\pi)$  bits of communication, which is optimal upto the factor of  $\log C_\pi$ . In addition, [4] identified another measure of information

---

\* Supported by the National Science Foundation under agreement CCF-1016565 and partially supported by CCF-1016509.

† Supported by an Alfred P. Sloan Fellowship, the National Science Foundation under agreement CCF-1016565, an NSF Career award, and by the Binational Science Foundation under agreement 2010089.





called the *internal* information cost of the protocol. This is the amount of information that is revealed by the parties of the protocol: let  $I_\pi^A$  denote the information revealed by the first party, and  $I_\pi^B$  denote the information revealed by the second party. Then the internal information cost is defined to be  $I_\pi = I_\pi^A + I_\pi^B$ . Since each of the parties already knows one of the inputs, the internal information cost is never more than the external information cost, with equality when the inputs to the parties are independent of each other. This quantity turns out to be the most interesting for applications to lower bounds. Indeed, Braverman and Rao [8] showed that the internal information cost required to compute a function is exactly equal to the amortized communication complexity of the function, so this quantity has a very natural interpretation, seemingly independent of information theory.

[4] showed that any protocol  $\pi$  can be simulated in  $O(\sqrt{I_\pi C_\pi} \log C_\pi)$  bits of communication. If we wish the simulation to not have a dependence on the communication of the original protocol, Braverman [6] showed that one can carry out the simulation using communication complexity  $2^{O(I_\pi)}$  (see also [9, 13]), a result that was subsequently proven to be tight by Ganor, Kol and Raz [11]. In addition, Braverman and Weinstein [9] (see also [13]) showed that if a function is computed by  $\pi$ , then the space of inputs must contain a nearly monochromatic rectangle of density  $2^{-O(I_\pi)}$ , a fact that can be used to prove lower bounds on the information complexity of computing functions.

In the setting of bounded round communication, Braverman and Rao [8] showed that a single message can be compressed to its internal information, giving a protocol that can simulate any  $r$ -round protocol with internal information cost  $I$  using  $I + O(r)$  bits of communication.

In this work, we generalize and strengthen several of these results, in the case that  $I^B \ll I^A, C$ . This case is interesting in part because many lower bounds for data structures involve proving lower bounds on so called *lopsided* problems, problems where the optimal lower bound on communication for one party is much smaller than for the other party (see [15], [18], [2], [17], [16], [5], [12], [19]). Indeed, our techniques allow us to reprove an important and well known theorem of Patrascu [18] giving a tight lower bound on the communication complexity of lopsided disjointness.

Our first theorem is somewhat analogous to the result of [4]. We show

► **Theorem 1.1.** *Every protocol  $\pi$  can be  $\epsilon$ -simulated by a protocol with expected communication  $O\left(I_\pi^A + \sqrt[4]{\|\pi\|^3} \cdot I_\pi^B \cdot \log(1/\epsilon) + \sqrt[4]{\|\pi\|^3} \cdot I_\pi^B \cdot \log \|\pi\| + \sqrt{\|\pi\|} \cdot I_\pi^B \cdot \log \|\pi\|\right)$ .*

We prove Theorem 1.1 in Section 3. At a high level, we first show that  $\pi$  can be simulated by a bounded round protocol, and then use the ideas of [8] to get the final simulation.

Our second result is an analogue of [6]:

► **Theorem 1.2.**  *$\pi$  can be simulated in communication complexity  $I_\pi^A \cdot 2^{O(I_\pi^B)}$ .*

Theorem 1.2 shows that when the information revealed by one of the parties is a constant, the communication is within a constant factor of the information. We prove Theorem 1.2 in Section 4. As a corollary to Theorem 1.2, we show the following,

► **Corollary 1.3.** *If  $\pi$  computes  $f(x, y)$ , then there exists a rectangle  $S \times T$  and a constant  $c$  such that*

$$\Pr[x \in S] \geq 2^{-O(I_\pi^A)}, \quad \Pr[y \in T | x \in S] \geq 2^{-O(I_\pi^B)}$$

and

$$\Pr[f(x, y) = c | (x, y) \in S \times T] \geq 2/3.$$

■ **Table 1** Known bounds on the complexity of simulating  $r$ -round protocols with communication  $C$  and information  $I^A, I^B$ .

Reference	Communication Complexity of the Simulation
[4]	$O\left(\sqrt{(I^A + I^B)C} \log C\right)$
[4]	$O\left((I^A + I^B) \log C\right)$ (when inputs are independent)
[6]	$2^{O(I^A + I^B)}$
[8]	$I^A + I^B + O\left(\sqrt{r \cdot (I^A + I^B)} + 1\right) + r \log(1/\epsilon)$
Theorem 1.1	$O\left(I_\pi^A + \sqrt[4]{\ \pi\ ^3 \cdot I_\pi^B} \cdot \log(1/\epsilon) + \sqrt[4]{\ \pi\ ^3 \cdot I_\pi^B} \cdot \log \ \pi\  + \sqrt{\ \pi\  \cdot I_\pi^B} \cdot \log \ \pi\ \right)$
Theorem 1.2	$I^A \cdot 2^{O(I^B)}$

One can view Corollary 1.3 as defining an asymmetric notion of discrepancy, and showing that if the information complexity is small, then the discrepancy must be large. Corollary 1.3 is a useful tool to prove lower bounds on lopsided problems. We illustrate this by using the ideas going into Corollary 1.3 to give optimal lower bounds on the communication complexity of lopsided disjointness (a bound first proved by Patrascu [18]).

Table 1 summarizes all of the simulation results discussed in this introduction.

## 2 Preliminaries

Unless otherwise stated, logarithms in this text are computed base two. Random variables are denoted by capital letters and values they attain are denoted by lower-case letters. For example,  $A$  may be a random variable and then  $a$  denotes a value  $A$  may attain and we may consider the event  $A = a$ . Given  $a = a_1, a_2, \dots, a_n$ , we write  $a_{\leq i}$  to denote  $a_1, \dots, a_i$ . We define  $a_{> i}$  and  $a_{\leq i}$  similarly.  $[\ell]$  denotes the set  $\{1, 2, \dots, \ell\}$ .

We use the notation  $p(a)$  to denote both the distribution on the variable  $a$ , and the number  $\Pr_p[A = a]$ . The meaning will be clear from context. We write  $p(a|b)$  to denote either the distribution of  $A$  conditioned on the event  $B = b$ , or the number  $\Pr[A = a|B = b]$ . Again, the meaning will be clear from context. Given a distribution  $p(a, b, c, d)$ , we write  $p(a, b, c)$  to denote the marginal distribution on the variables  $a, b, c$  (or the corresponding probability). We often write  $p(ab)$  instead of  $p(a, b)$  for conciseness of notation. If  $W$  is an event, we write  $p(W)$  to denote its probability according to  $p$ . We denote by  $\mathbb{E}_{p(a)}[g(a)]$  the expected value of  $g(a)$  with respect to  $a$  distributed according to  $p$ .

For two distributions  $p, q$ , we write  $|p(a) - q(a)|$  to denote the  $\ell_1$  distance between the distributions  $p$  and  $q$ . We write  $p \stackrel{\epsilon}{\approx} q$  if  $|p - q| \leq \epsilon$ .

► **Proposition 2.1.** *Let  $p(x), q(x)$  be two distributions and  $F$  be an event such that  $p(x|F) \stackrel{\epsilon}{\approx} q(x)$ . Then if  $p(F) \geq 1 - \gamma$ , we have  $p(x) \stackrel{\epsilon + 2\gamma}{\approx} q(x)$ .*

**Proof.** The  $\ell_1$  distance between  $p, q$  can be expressed as  $2 \max_T (p(T) - q(T))$ , where the maximum is taken over all subsets of the support of  $p(x)$ . Let  $T$  be the maximizer. Then

$$\begin{aligned}
 |p(x) - q(x)| &= 2(p(T) - q(T)) \\
 &= (2(p(T|F)p(F) + p(\neg F)p(T|\neg F)) - q(T)) \\
 &\leq 2(p(T|F) - q(T)) + 2p(\neg F) \\
 &\leq \epsilon + 2\gamma,
 \end{aligned}$$

as required. ◀

► **Proposition 2.2.** Let  $p(x), q(x)$  be two distributions and  $F$  be an event such that  $p(x) \stackrel{\epsilon}{\approx} q(x)$ . Then if  $p(F) \geq 1 - \gamma \geq 3/4$ , we have  $p(x|F) \stackrel{\epsilon+4\gamma}{\approx} q(x)$ .

**Proof.** The  $\ell_1$  distance between  $p(x|F), q(x)$  can be expressed as  $2 \max_T p(T|F) - q(T)$ , where the maximum is taken over all subsets of the support of  $p(x)$ . Let  $T$  be the maximizer. Then

$$\begin{aligned} |p(x|F) - q(x)| &= 2(p(T|F) - q(T)) \\ &= 2(p(T, F)/p(F) - q(T)) \\ &\leq 2(p(T)/p(F) - q(T)) \\ &\leq 2(p(T)/(1 - \gamma) - q(T)) \\ &\leq 2(p(T)(1 + 2\gamma) - q(T)) \\ &\leq \epsilon + 4\gamma, \end{aligned}$$

as required. ◀

The *entropy* of a random variable  $A$ , conditioned on  $B$  is defined to be

$$H_p(A|B) = \sum_{a,b} p(ab) \log \frac{1}{p(a|b)}$$

For a binary random variable  $A$ , we denote the entropy of  $A$  to be

$$h(p(0)) = -[p(0) \log p(0) + (1 - p(0)) \log(1 - p(0))]$$

The *divergence* between two distributions is defined to be

$$\mathbb{D} \left( \frac{p}{q} \right) = \sum_a p(a) \log \frac{p(a)}{q(a)}$$

The *mutual information* between two random variables  $A, B$ , conditioned on  $C$  is defined to be

$$I_p(A; B|C) = \sum_{a,b,c} p(abc) \log \frac{p(abc)}{p(a|c)p(b|c)}.$$

This is always a non-negative quantity, and is at most  $\log |\text{Supp}(A)|$ . When the underlying distribution  $p$  is clear from the context, we sometimes omit it from the notation. The mutual information satisfies the chain rule:

► **Proposition 2.3** (Chain Rule).  $I(A_1 A_2; B|C) = I(A_1; B|C) + I(A_2; B|A_1 C)$ .

Pinsker's inequality bounds the  $\ell_1$  distance in terms of the divergence:

► **Proposition 2.4** (Pinsker).  $\mathbb{D} \left( \frac{p}{q} \right) \geq |p - q|^2$ .

An alternate formulation is as follows:

► **Proposition 2.5** (Alternate Pinsker).  $\mathbb{E}_{p(bc)} [|p(a|bc) - p(a|c)|] \leq \sqrt{I(A; B|C)}$ .

The chain rule easily gives the following inequality:

► **Proposition 2.6** (Data Processing Inequality). *If the random variable  $A$  determines  $B$ , then  $I(A; C) \geq I(B; C)$ .*

► **Proposition 2.7** ([11]). *Let  $p(ab)$  be a distribution and  $q(a)$  be another. Then*

$$\mathbb{E}_{p(b)} \left[ \mathbb{D} \left( \frac{p(a|b)}{p(a)} \right) \right] \leq \mathbb{E}_{p(b)} \left[ \mathbb{D} \left( \frac{p(a|b)}{q(a)} \right) \right].$$

We shall sometimes deal with distribution on strings of variable length. We have the following proposition, which follows from Shannon's source coding theorem:

► **Proposition 2.8.** *Suppose  $A$  is a random variable supported on binary strings of length up to  $n$ , such that no string in the support of  $A$  is a prefix of another string in the support of  $A$ . Then  $I(A; B|C) \leq \mathbb{E}[|A|]$ .*

## 2.1 Communication Complexity

For a more involved introduction to communication complexity, we refer the reader to the book [14]. Given a protocol  $\pi$  that operates on inputs  $x, y$  drawn from a distribution  $\mu$  using public randomness<sup>1</sup>  $r$  and messages  $m$ , we write  $\pi(xymr)$  to denote the joint distribution of these variables. We write  $\|\pi\|$  to denote the *communication complexity* of  $\pi$ , namely the maximum number of bits that may be exchanged by the protocol in any execution. The maximum number of alternations between messages sent by Alice and those sent by Bob is called the number of *rounds* of the protocol.

Let  $q(x, y, a)$  be an arbitrary distribution. We say that a protocol  $\pi$   $\delta$ -simulates  $q$ , if there is a function  $g$  and a function  $h$  such that

$$\pi(x, y, g(x, r, m), h(y, r, m)) \stackrel{\delta}{\approx} q(x, y, a, a),$$

where  $q(x, y, a, a)$  is the distribution on 4-tuples  $(x, y, a, a)$  where  $(x, y, a)$  are distributed according to  $q$ . Thus if  $\pi$   $\delta$ -simulates  $q$ , the protocol allows the parties to sample  $a$  according to  $q(a|xy)$ .

If  $\lambda$  is a protocol with inputs  $x, y$ , public randomness  $r'$  and messages  $m'$ , we say that  $\pi$   $\delta$ -simulates  $\lambda$  if  $\pi$   $\delta$ -simulates  $\lambda(x, y, (r', m'))$ . We say that  $\pi$  simulates  $\lambda$  if  $\pi$  0-simulates  $\lambda$ .

We say that  $\pi$  computes

a function  $f(x, y)$  with success probability  $1 - \delta$ , if  $\pi$   $\delta$ -simulates  $\pi(x, y, f(x, y))$ .

We shall sometimes refer to the *expected communication*, or *expected number of rounds* of a protocol  $\pi$ . We note here that one can always use a bound on the expected communication or number of rounds to get a bound on the worst case communication, via the following proposition:

► **Proposition 2.9.** *If  $\pi$  has expected communication  $c$ , then it can be  $\gamma$ -simulated by a protocol with communication  $c/\gamma$ .*

Our work relies on ways to measure the information complexity of a protocol (see [4, 7] and references within for a more detailed overview). The *internal information cost* [4] of  $\pi$  is

<sup>1</sup> In our paper we define protocols where the public randomness is sampled from a continuous (i.e. non-discrete) set. Nevertheless, we often treat the randomness as if it were supported on a discrete set, for example by taking the sum over the set rather than the integral. This simplifies notation throughout our proofs, and does not affect correctness in any way, since all of our public randomness can be approximated to arbitrary accuracy by sufficiently dense finite sets.

defined to be  $I_\pi(X; M|YR) + I_\pi(Y; M|XR)$ . This quantity is the sum of the information learnt by Alice about Bob's input,  $I_\pi^B = I_\pi(Y; M|XR)$ , and the information learnt by Bob about Alice's input,  $I_\pi^A = I_\pi(X; M|YR)$ . We will sometimes say that the internal information is  $(I^A, I^B)$  when we want to consider the values of both quantities instead of the sum.

### 2.1.1 Results from prior work

► **Theorem 2.10** ([8]). *For every  $\epsilon > 0$ , if  $\pi$  is an a protocol with internal information cost  $I$  and  $r$  rounds in expectation, then  $\pi$  can be  $\epsilon$ -simulated with expected communication  $I + O(\sqrt{r \cdot I} + 1) + r \log(1/\epsilon)$ .*

► **Theorem 2.11** ([8]). *For any  $f, \mu, \epsilon$ , let  $\pi$  be the protocol computing  $f$  on  $n$  independent pairs of inputs, each drawn from the distribution  $\mu$  and probability of error is at most  $\epsilon$  on each pair, then there exists a protocol  $\tau$  computing  $f$  on a single input pair with communication  $\|\tau\| = \|\pi\|$  and information  $I_\tau^A \leq \frac{I_\pi^A}{n}$ ,  $I_\tau^B \leq \frac{I_\pi^B}{n}$*

## 3 Compressing Protocols with Asymmetric Information – I

In this section, we show how to compress protocols to take advantage of situations where the information learnt by one party is significantly larger than the information revealed by the other party. We shall prove Theorem 1.1.

We compress the given protocol in two steps. In the first step, we convert the protocol into a bounded round protocol, while controlling its internal information cost. In the second step, we apply Theorem 2.10 to conclude the proof. The first step is captured by the following theorem:

► **Theorem 3.1** (Bounded round simulation). *Given any protocol  $\pi$  and a parameter  $k$ , there exists a protocol that simulates  $\pi$  with  $\sqrt{I_\pi^B} \cdot \|\pi\| + \|\pi\|/k$  number of rounds in expectation, and internal information at most  $\frac{\|\pi\| \log \|\pi\|}{k} + k\sqrt{I_\pi^B} \cdot \|\pi\| + I_\pi^A + 2 \log \|\pi\| \sqrt{\|\pi\| \cdot I_\pi^B} + 3$ .*

We use the protocol  $\tau$  given in figure 1<sup>2</sup> to simulate  $\pi$ .

Let  $M$  denote the output of  $\tau$ . Then we claim that the distribution of  $m$  is correct:

► **Lemma 3.2.**  $\tau(xyrm) = \pi(xyrm)$ .

**Proof.** It is clear that  $\tau(xyr) = \pi(xyr)$ . For each  $i \in [\|\pi\|]$ , if  $m_i$  is to be sent by Alice in  $\pi$ , then  $m_i = 1$  exactly when  $\rho_i < \pi(m_i|xrm_{<i})$ , and if  $m_i$  is to be sent by Bob in  $\pi$ , then  $m_i = 1$  exactly when  $\rho_i < \pi(m_i|yrm_{<i})$ . Thus, if  $m_i$  is to be sent by Alice in  $\pi$ ,  $\tau(m_i|xyrm_{<i}) = \pi(m_i|xrm_{<i}) = \pi(m_i|xyrm_{\leq i})$ . On the other hand, if  $m_i$  is to be sent by Bob in  $\pi$ , then  $\tau(m_i|xyrm_{<i}) = \pi(m_i|yrm_{<i}) = \pi(m_i|xyrm_{<i})$ . Thus

$$\tau(xyrm) = \tau(xyr) \cdot \prod_{i=1}^{\|\pi\|} \tau(m_i|xyrm_{<i}) = \pi(xyr) \cdot \prod_{i=1}^{\|\pi\|} \pi(m_i|xyrm_{<i}) = \pi(xyrm).$$

Let  $L$  denote the number of *mistake* indices  $j$  reported to Alice by Bob in  $\tau$ . Then we have:

<sup>2</sup> Here we do not bother optimizing the communication of  $\tau$ , since the communication will be eventually optimized via Theorem 2.10.

**Input:**  $x, y$ , the inputs to  $\pi$ . A parameter  $k$ .  
**Output:**  $m, r$  distributed according to  $\pi(mr|xy)$ .  
**Public Randomness:** The public randomness  $r$  of  $\pi$ , as well as an additional sequence of uniformly random numbers  
 $r' = \rho_1, \dots, \rho_{\|\pi\|} \in [0, 1]$ .

Let  $m$  be the empty string;  
**while**  $|m| < \|\pi\|$  **do**  
    Set  $t = |m|$ ;  
    **for**  $i = t + 1, \dots, \min\{k + t, \|\pi\|\}$  **do**  
        **if**  $m_i$  is sent by Bob in  $\pi$  **then** Alice checks if  $\rho_i < \pi(m_i = 1|xrm_{<i})$ , and sets  $m_i = 1$  if this is the case. Otherwise she sets  $m_i = 0$ ;  
        **else** Alice samples  $m_i$  privately according to the distribution  $\pi(m_i|xrm_{<i})$ ;  
    **end**  
    Alice sends the current transcript  $m$  to Bob;  
    Bob computes the smallest index  $j \in [\min\{k + t, \|\pi\|\}]$  such that  $m_j$  would have been sent by Bob in  $\pi$  and  $\rho_j$  lies in the interval between  $\pi(m_j = 1|xrm_{<j})$  and  $\pi(m_j = 1|yrm_{<j})$ . Bob can check this using  $\rho_j, m, y, r$ ;  
    Bob sends  $j$  to Alice, or reports that there is no such  $j$ ;  
    Alice corrects  $m$  if such  $j$  is found, by flipping the bit  $m_j$ , and truncating  $m = m_{\leq j}$ ;  
**end**  
**return**  $m$ ;

■ **Figure 1** Protocol  $\tau$  simulating  $\pi$ .

► **Lemma 3.3.** *The number of rounds in  $\tau$  is at most  $\|\pi\|/k + L = \sqrt[4]{\|\pi\|^3 \cdot I_\pi^B} + L$ .*

**Proof.** There are  $L$  rounds where Alice needs to truncate  $m$ . In every other round, at least  $k$  new bits of the messages of  $\pi$  are sampled, so there can be at most  $\|\pi\|/k$  additional rounds of communication. ◀

Given the last lemma, we can bound the number of rounds in the protocol by bounding  $L$ :

► **Lemma 3.4.**  $\mathbb{E}[L] \leq \sqrt{I_\pi^B \cdot \|\pi\|}$ .

**Proof.** Let  $L_i$  denote the indicator random variable for the event that the  $i$ 'th index of the message is corrected by Bob in  $\tau$ , so  $L = \sum_{i=1}^{\|\pi\|} L_i$ . If the  $i$ 'th message is sent by Alice, then  $L_i = 0$ . On the other hand, if it is sent by Bob, by Proposition 2.5, we get

$$\begin{aligned} \tau(L_i = 1) &= \mathbb{E}_{xyrm_{<i}} [|\pi(m_i|xrm_{<i}) - \pi(m_i|yrm_{<i})|] \\ &= \mathbb{E}_{xyrm_{<i}} [|\pi(m_i|xrm_{<i}) - \pi(m_i|xyrm_{<i})|] \\ &\leq \sqrt{I(M_i; Y|XRM_{<i})}. \end{aligned}$$

The penultimate inequality is true since  $m_i$  is sampled by Bob in  $\pi$ , hence is independent of  $x$  conditioned on  $y, r, m_{<i}$ .

Thus by linearity of expectation and the Cauchy Schwartz inequality, we get

$$\begin{aligned} \mathbb{E}[L] &\leq \sum_{i=1}^{\|\pi\|} \sqrt{I(M_i; Y | XRM_{<i})} \\ &\leq \sqrt{\|\pi\| \cdot \sum_{i=1}^{\|\pi\|} I(M_i; Y | XRM_{<i})} \\ &= \sqrt{\|\pi\| \cdot I(M; Y | XR)}, \end{aligned}$$

where here we used the chain rule (Proposition 2.3) in the last step.  $\blacktriangleleft$

Lemma 3.4 and Lemma 3.3 together imply that the expected number of rounds in  $\tau$  is at most  $\sqrt{I_\pi^B \cdot \|\pi\|} + \|\pi\|/k$ . It only remains to bound the internal information of  $\tau$ :

► **Lemma 3.5.** *The internal information of  $\tau$  is at most*

$$\frac{\|\pi\| \log \|\pi\|}{k} + k\sqrt{I_\pi^B \cdot \|\pi\|} + I_\pi^A + 2 \log \|\pi\| \sqrt{\|\pi\| \cdot I_\pi^B} + 3$$

**Proof.** Recall that  $R'$  denotes the sequence of numbers  $\rho_1, \dots, \rho_{\|\pi\|}$ . The public randomness of  $\tau$  consists of  $R, R'$ . Let  $Z$  denote the messages exchanged in the protocol  $\tau$ . Let  $Z_A$  denote the bits sent by Alice, and  $Z_B$  denote the bits sent by Bob. Then the information learnt by Alice can be expressed as

$$I_\tau(Z_A Z_B; Y | XRR') = I_\tau(Z_B; Y | XRR') + I_\tau(Z_A; Y | XRR' Z_B), \quad (1)$$

by the chain rule. The the second term of (1) is 0, since Alice's messages are independent of  $Y$ , given Bob's messages and Alice's inputs. For the first term, we use Proposition 2.8 to bound it by  $\mathbb{E}[|Z_B|]$ . The total number of rounds of the protocol is at most  $L + \|\pi\|/k$ , since every round where there is no mistake must simulate at least  $k$  messages from the protocol. Thus  $\mathbb{E}[|Z_B|] \leq (\mathbb{E}[L] + \|\pi\|/k) \log \|\pi\| \leq \sqrt{I_\pi^B \cdot \|\pi\|} \log \|\pi\| + \frac{\|\pi\| \log \|\pi\|}{k}$ , by Lemma 3.4.

Next we bound the information learnt by Bob in  $\tau$ . This can be written

$$I_\tau(Z; X | YRR') = \sum_{i=1}^{|Z|} I_\tau(Z_i; X | Z_{<i} YRR') = \mathbb{E}_{xyrr'z} \left[ \sum_{i=1}^{|z|} \mathbb{D} \left( \frac{z_i | xyrr' z_{<i}}{z_i | yrr' z_{<i}} \right) \right] \quad (2)$$

by the chain rule.

► **Claim 3.6.** *For  $x \in [0, 1/2]$   $\log(1/(1-x)) \leq 3x$*

**Proof.** Let  $T = \ln(1/(1-x))$ . The Taylor expansion of  $\ln(1/(1-x))$  gives,

$$\begin{aligned} T = \ln(1/(1-x)) &= x + x^2/2 + x^3/3 + \dots \\ &= x + x(x/2 + x^2/3 + x^3/4 + \dots) \\ &\leq x + x \cdot T. \end{aligned}$$

This implies,  $T \leq x/(1-x)$ . Since,  $x \leq 1/2$ ,  $T \leq 2x$ . Now,

$$\log(1/(1-x)) = \ln(1/(1-x))/\ln(2) \leq 1.5 \ln(1/(1-x)) \leq 3x,$$

which follows from the fact that  $1/\ln(2) \leq 1.5$   $\blacktriangleleft$

► **Claim 3.7.** *If  $m_{<j}$  represents the messages of  $\pi$  sampled by the simulation  $\tau$  at the point the messages  $z_{<i}$  were sent, then*

$$\mathbb{E}_{r'|xyrz_{<i}} \left[ \mathbb{D} \left( \frac{\tau(z_i|xyrr'z_{<i})}{\tau(z_i|yrr'z_{<i})} \right) \right] \begin{cases} = 0 & \text{if Bob sends } z_i, \\ \leq 1 & \text{if } z_i \text{ is discarded,} \\ \leq \mathbb{D} \left( \frac{\pi(m_j|xyrm_{<j})}{\pi(m_j|yrm_{<j})} \right) & \text{if Alice sends } m_j, \\ \leq \sqrt{\mathbb{D} \left( \frac{\pi(m_j|xyrm_{<j})}{\pi(m_j|xrm_{<j})} \right)} \log \|\pi\| + \frac{3}{\|\pi\|} & \text{if Bob sends } m_j. \end{cases}$$

**Proof.** When  $z_i$  is sent by Bob, both distributions are the same, so the divergence is 0.

To prove the remaining cases, we apply Proposition 2.7. When  $z_i$  is to be discarded, set  $q(z_i|yrr'z_{<i})$  to be the uniform distribution on bits. When  $q$  is the uniform distribution on the bits,  $\mathbb{D} \left( \frac{p}{q} \right) = 1 - h(p/0) \leq 1$ . Since  $\mathbb{D} \left( \frac{p}{q} \right) \leq 1$  for any  $p$ , the bound follows using Proposition 2.7. When  $m_j$  is sent by Alice in  $\pi$ , observe that the distribution of  $\tau(z_i|xyrr'z_{<i})$  is exactly the same as the distribution of  $\pi(m_j|xyrm_{<j})$ . Set  $q(z_i|yrr'z_{<i}) = \pi(m_j|yrm_{<j})$ . The bound follows by Proposition 2.7. For the last case, observe that in  $\tau$ ,  $z_i$  is determined by  $xyrr'm_{<j}$ , since  $z_i = 1$  exactly when  $\rho_i < \pi(m_j = 1|xrm_{<j})$ . Set

$$b(\rho_i, y, r, r', m_{<j}) = \begin{cases} 1 & \text{if } \rho_i < \pi(m_j = 1|yrm_{<j}), \\ 0 & \text{otherwise} \end{cases}$$

and

$$q(z_i|yrr'z_{<i}) = \begin{cases} 1 - 1/\|\pi\| & \text{if } b(\rho_i, y, r, r', m_{<j}) = z_i, \\ 1/\|\pi\| & \text{otherwise.} \end{cases}$$

When  $\rho_i$  is in between  $\pi(m_j = 1|yrm_{<j})$  and  $\pi(m_j = 1|xrm_{<j})$ ,

$$\mathbb{D} \left( \frac{\tau(z_i|xyrr'z_{<i})}{q(z_i|yrr'z_{<i})} \right) = \log(1/(1/\|\pi\|)) = \log \|\pi\|,$$

for all  $z_i$  with positive probability. When  $\rho_i$  is not in between those two quantities,

$$\mathbb{D} \left( \frac{\tau(z_i|xyrr'z_{<i})}{q(z_i|yrr'z_{<i})} \right) \leq \log \frac{1}{1 - 1/\|\pi\|} \leq 3/\|\pi\|.$$

which follows from Claim 3.6 and the assumption that  $\|\pi\| > 2$ . It is safe to assume that  $\|\pi\| > 2$ , as the compression is trivial for protocols with communication at most 2.

The probability of the first case is at most  $\sqrt{\mathbb{D} \left( \frac{\pi(m_j|xyrm_{<j})}{\pi(m_j|yrm_{<j})} \right)}$ , by Proposition 2.4. ◀

Now given  $Z$ , call  $i$  good if the message  $Z_i$  does not correspond to a mistake and is not



discarded by the simulation. Let  $G$  denote the set of good indices. We have,

$$\begin{aligned} & \mathbb{E}_{xyrr'z} \left[ \sum_{i=1}^{|z|} \mathbb{D} \left( \frac{\tau(z_i|xyrr'z_{<i})}{\tau(z_i|yrr'z_{<i})} \right) \right] \\ &= \mathbb{E}_{xyrz} \left[ \sum_{i=1}^{|z|} \mathbb{E}_{r'|xyrz_{<i}} \left[ \mathbb{D} \left( \frac{\tau(z_i|xyrr'z_{<i})}{\tau(z_i|yrr'z_{<i})} \right) \right] \right] \\ &= \mathbb{E}_{xyrz} \left[ \sum_{i \in G} \mathbb{E}_{r'|xyrz_{<i}} \left[ \mathbb{D} \left( \frac{\tau(z_i|xyrr'z_{<i})}{\tau(z_i|yrr'z_{<i})} \right) \right] \right] \\ & \quad + \mathbb{E}_{xyrz} \left[ \sum_{i \notin G} \mathbb{E}_{r'|xyrz_{<i}} \left[ \mathbb{D} \left( \frac{\tau(z_i|xyrr'z_{<i})}{\tau(z_i|yrr'z_{<i})} \right) \right] \right] \end{aligned}$$

For every  $Z$ , at most  $k \cdot L$  indices are discarded. This is because, at most  $k$  indices are discarded for every round with a mistake. Then by Claim 3.7,

$$\mathbb{E}_{xyrz} \left[ \sum_{i \notin G} \mathbb{E}_{r'|xyrz_{<i}} \left[ \mathbb{D} \left( \frac{\tau(z_i|xyrr'z_{<i})}{\tau(z_i|yrr'z_{<i})} \right) \right] \right] \leq k \mathbb{E}[L]$$

For every index  $i \in G$ , by Claim 3.7

$$\begin{aligned} & \mathbb{E}_{xyrz} \left[ \sum_{i \in G} \mathbb{E}_{r'|xyrz_{<i}} \left[ \mathbb{D} \left( \frac{\tau(z_i|xyrr'z_{<i})}{\tau(z_i|yrr'z_{<i})} \right) \right] \right] \\ & \leq \mathbb{E}_{\pi(mxyr)} \left[ \mathbb{D} \left( \frac{\pi(m_j|xym_{<j})}{\pi(m_j|yrm_{<j})} \right) \right] \\ & \quad + \mathbb{E}_{\pi(mxyr)} \left[ \sum_{j=1}^{\|\pi\|} \sqrt{\mathbb{D} \left( \frac{\pi(m_j|xym_{<j})}{\pi(m_j|xrm_{<j})} \right)} \log \|\pi\| + \frac{3}{\|\pi\|} \right] \\ & = \mathbb{E}_{\pi(mxyr)} \left[ \mathbb{D} \left( \frac{\pi(m_j|xym_{<j})}{\pi(m_j|yrm_{<j})} \right) \right] + \log \|\pi\| \mathbb{E}_{\pi(mxyr)} \left[ \sum_{j=1}^{\|\pi\|} \sqrt{\mathbb{D} \left( \frac{\pi(m_j|xyr)}{\pi(m_j|xr)} \right)} \right] + 3 \\ & \leq \mathbb{E}_{\pi(mxyr)} \left[ \mathbb{D} \left( \frac{\pi(m_j|xym_{<j})}{\pi(m_j|yrm_{<j})} \right) \right] \\ & \quad + \log \|\pi\| \sqrt{\|\pi\| \cdot \mathbb{E}_{\pi(mxyr)} \left[ \sum_{j=1}^{\|\pi\|} \mathbb{D} \left( \frac{\pi(m_j|xyr)}{\pi(m_j|xr)} \right) \right]} + 3 \\ & = I_\pi^A + \log \|\pi\| \sqrt{\|\pi\| \cdot I_\pi^B} + 3, \end{aligned}$$

where the penultimate inequality follows from an application of Cauchy Schwartz inequality, and the last inequality follows from the definition of  $I_\pi^A, I_\pi^B$ .  $\blacktriangleleft$

► **Corollary 3.8.** *Given any protocol  $\pi$ , there exists a protocol that simulates  $\pi$  with  $2 \cdot \sqrt[4]{\|\pi\|^3} \cdot I_\pi^B$  number of rounds in expectation, and internal information at most  $I_\pi^A + \sqrt[4]{\|\pi\|^3} \cdot I_\pi^B + \sqrt[4]{\|\pi\|^3} \cdot I_\pi^B \cdot \log \|\pi\| + 2 \cdot \sqrt{\|\pi\| \cdot I_\pi^B} \cdot \log \|\pi\| + 3$ .*

**Proof.** Set  $k = \sqrt[4]{\|\pi\|/I_\pi^B}$ . By Theorem 3.1, we get that the expected number of rounds of the simulation is at most  $\sqrt{I_\pi^B \cdot \|\pi\|} + \sqrt[4]{\|\pi\|^3} \cdot I_\pi^B \leq 2 \sqrt[4]{\|\pi\|^3} \cdot I_\pi^B$  (since  $I_\pi^B \leq \|\pi\|$ ) and

Simulation $\pi$
<p><b>Public randomness:</b> A sequence of <math>L = 10 \cdot  U  \cdot 2^{20(I^A+1)}</math> tuples <math>(z_i, a_i, b_i) \in U \times [0, 2^{20(I^A+1)}] \times [0, 2^{20(I^B+1)}]</math>, for <math>i = 1, 2, \dots, L</math>, and a random function <math>h : [L] \rightarrow [2^{40(I^A+1)}]</math>.</p> <ol style="list-style-type: none"> <li>1. Alice computes the set <math>\mathcal{A} = \left\{ i \mid a_i \leq p_A(z_i), b_i \leq p_B(z_i) \cdot 2^{20(I^B+1)} \right\}</math>, and Bob computes the set <math>\mathcal{B} = \left\{ i \mid a_i \leq q_A(z_i) \cdot 2^{20(I^A+1)}, b_i \leq q_B(z_i) \right\}</math>.</li> <li>2. Alice computes <math>i^*</math>, the smallest element of <math>\mathcal{A}</math>.</li> <li>3. Alice sends <math>h(i^*)</math> to Bob.</li> <li>4. If there is a unique <math>i \in \mathcal{B}</math> such that <math>h(i) = h(i^*)</math>, Bob accepts and assumes that the outcome of the protocol is <math>z_i</math>. Otherwise Bob aborts.</li> </ol>

■ **Figure 2** The sampling procedure.

the internal information is at most  $I_\pi^A + \sqrt[4]{\|\pi\|^3 \cdot I_\pi^B} + \sqrt[4]{\|\pi\|^3 \cdot I_\pi^B} \cdot \log \|\pi\| + 2 \cdot \sqrt{\|\pi\| \cdot I_\pi^B} \cdot \log \|\pi\| + 3$  ◀

Applying Theorem 2.10 to the simulation guaranteed by Corollary 3.8, gives a simulation with communication bounded by

$$O\left(I_\pi^A + \sqrt[4]{\|\pi\|^3 \cdot I_\pi^B} \cdot \log(1/\epsilon) + \sqrt[4]{\|\pi\|^3 \cdot I_\pi^B} \cdot \log \|\pi\| + \sqrt{\|\pi\| \cdot I_\pi^B} \cdot \log \|\pi\|\right),$$

as required in Theorem 1.1.

## 4

 Compressing Protocols with Asymmetric Information – II

In this section, we prove Theorem 1.2, (i.e) we show how to simulate  $\pi$  with communication  $I_\pi^A \cdot 2^{O(I_\pi^B)}$ .

► **Theorem 4.1.** *Let  $U$  be a finite set. Let  $p_A, p_B, q_A, q_B : U \rightarrow [0, 1]$  be such that  $\forall z \in U$ ,  $\mu(z) = p_A(z)q_B(z)$ ,  $p(z) = p_A(z)p_B(z)$  and  $q(z) = q_A(z)q_B(z)$  are distributions. There exists a randomized protocol with inputs  $p_A, p_B$  to Alice and  $q_A, q_B$  to Bob, such that*

■ *Both Alice and Bob either accept and compute (possibly different) samples  $z \in U$ , or abort the protocol.*

■  $\Pr[\text{Both parties accept}] \geq 2^{-O\left(\mathbb{D}\left(\frac{\mu}{q}\right)_{+1}\right)}$ .

■ *Given that both parties accept, the distribution of their samples is 0.35-close in  $\ell_1$  distance to the distribution where both parties sample the same sample from  $\mu(z)$ .*

Let  $I^B = \mathbb{D}\left(\frac{\mu}{p}\right)$  and  $I^A = \mathbb{D}\left(\frac{\mu}{q}\right)$ . Figure 2 describes the randomized protocol promised by the lemma. Let

$$\mathcal{G} = \left\{ z \mid 2^{20(I^B+1)} \cdot p(z) \geq \mu(z), 2^{20(I^A+1)} \cdot q(z) \geq \mu(z) \right\}.$$

We need the following simple claim, which was proved in [6].

► **Claim 4.2.**  $\mu(\mathcal{G}) \geq \frac{9}{10}$ .

We proceed with the analysis of the simulation.

► **Lemma 4.3.**  $\Pr[i^* \text{ is defined}] \geq 1 - e^{-11}$ .

**Proof.** For each  $i$ , we have

$$\begin{aligned} \Pr[i \in \mathcal{A}] &= \sum_{z \in U} \frac{p_A(z)p_B(z)}{|U| \cdot 2^{20(I^A+1)}} \\ &= \frac{1}{|U| \cdot 2^{20(I^A+1)}} \sum_{z \in U} p_A(z)p_B(z) \\ &= \frac{1}{|U| \cdot 2^{20(I^A+1)}}. \end{aligned} \quad (\text{since } p \text{ is a distribution})$$

The probability that  $i^*$  is not defined is exactly equal to the probability that  $i \notin \mathcal{A}$  for all  $1 \leq i \leq L$ . Thus,

$$\begin{aligned} \Pr[i^* \text{ not defined}] &= \left(1 - \frac{1}{|U| \cdot 2^{20(I^A+1)}}\right)^L \\ &\leq e^{-L/|U| \cdot 2^{20(I^A+1)}}. \quad (\text{using } (1-x)^n \leq e^{-xn}, x \geq 0) \\ &\leq e^{-10}. \end{aligned}$$

► **Lemma 4.4.** For  $z \in U$ ,

$$\Pr[z_{i^*} = z \ \& \ i^* \in \mathcal{B} | i^* \text{ is defined}] \leq \frac{\mu(z)}{2^{20(I^B+1)}},$$

with equality when  $z \in \mathcal{G}$ .

**Proof.**

$$\Pr[z_{i^*} = z \ \& \ i^* \in \mathcal{B} | i^* \text{ is defined}] = \Pr[z_{i^*} = z | i^* \text{ is defined}] \cdot \Pr[i^* \in \mathcal{B} | z_{i^*} = z]. \quad (3)$$

We have

$$\begin{aligned} \Pr[z_{i^*} = z | i^* \text{ is defined}] &= \frac{p_A(z)p_B(z)2^{-20(I^A+1)}}{\sum_{z \in U} p_A(z)p_B(z)2^{-20(I^A+1)}} \\ &= p_A(z)p_B(z). \end{aligned} \quad (4)$$

Let us now analyze  $\Pr[i^* \in \mathcal{B} | z_{i^*} = z]$ . If  $z_{i^*} = z$ , we have  $a_{i^*} \leq p_A(z), b_{i^*} \leq p_B(z)2^{20(I^B+1)}$ . Thus  $\Pr[i^* \in \mathcal{B} | z_{i^*} = z]$  is exactly

$$\begin{aligned} \Pr[i^* \in \mathcal{B} | z_{i^*} = z] &= \min \left\{ \frac{q_B(z)}{2^{20(I^B+1)}p_B(z)}, 1 \right\} \cdot \min \left\{ \frac{2^{20(I^A+1)}q_A(z)}{p_A(z)}, 1 \right\} \\ &\leq \frac{q_B(z)}{2^{20(I^B+1)}p_B(z)}. \end{aligned} \quad (5)$$

Equality holds in (5) when  $z \in \mathcal{G}$ , since for such  $z$ ,  $\frac{q_B(z)}{2^{20(I^B+1)}p_B(z)} \leq 1$  and  $\frac{2^{20(I^A+1)}q_A(z)}{p_A(z)} \geq 1$ . Therefore, using (4),(5),(3)

$$\Pr[z_{i^*} = z \ \& \ i^* \in \mathcal{B}] \leq p_A(z)p_B(z) \cdot \frac{q_B(z)}{2^{20(I^B+1)}p_B(z)} = \mu(z)/2^{20(I^B+1)},$$

with equality for  $z \in \mathcal{G}$ .

When  $i^*$  is defined, let  $E$  denote the event that  $i^*$  is the only possible index that is in  $\mathcal{B}$  and consistent with the message Bob receives, namely:  $\forall i \neq i^*, i \in \mathcal{B} \Rightarrow h(i) \neq h(i^*)$ . Then we have

► **Lemma 4.5.**  $\Pr[\neg E | i^* \text{ is defined}] \leq \frac{L2^{-40(I^A+1)-20(I^B+1)}}{|U| \Pr[i^* \text{ is defined}]}$ .

**Proof.** Given that  $i^*$  is defined, probability that  $i \in \mathcal{B}$  and  $h(i) = h(i^*)$  is at most  $\frac{\Pr[i \in \mathcal{B}] \cdot 2^{-40(I^A+1)}}{\Pr[i^* \text{ is defined}]}$ . Thus,

$$\begin{aligned} \frac{\Pr[i \in \mathcal{B}] \cdot 2^{-40(I^A+1)}}{\Pr[i^* \text{ is defined}]} &= \frac{1}{|U| \cdot \Pr[i^* \text{ is defined}]} \cdot \sum_{z \in U} q_A(z) \cdot 2^{-20(I^B+1)} q_B(z) 2^{-40(I^A+1)} \\ &\leq \frac{2^{-40(I^A+1)-20(I^B+1)}}{|U| \cdot \Pr[i^* \text{ is defined}]} \end{aligned}$$

Thus, by the union bound, the probability than any such  $i$  is accepted by Bob is at most  $\frac{L2^{-40(I^A+1)-20(I^B+1)}}{|U| \Pr[i^* \text{ is defined}]}$  ◀

► **Lemma 4.6.**  $\Pr[i^* \in \mathcal{B} | i^* \text{ is defined}] \geq \mu(\mathcal{G}) \cdot 2^{-20(I^B+1)}$

**Proof.**

$$\begin{aligned} \Pr[i^* \in \mathcal{B} | i^* \text{ is defined}] &= \sum_{z \in |U|} \Pr[z_{i^*} = z, i^* \in \mathcal{B} | i^* \text{ is defined}] \\ &\geq \sum_{z \in \mathcal{G}} \mu(z) / 2^{20(I^B+1)} && \text{(by Lemma 4.4)} \\ &= \mu(\mathcal{G}) \cdot 2^{-20(I^B+1)}. && \text{(by Lemma 4.3)} \end{aligned}$$

► **Lemma 4.7.**  $\Pr[\text{Both parties accept} | i^* \text{ is defined}] \geq \frac{8}{10} \cdot 2^{-20(I^B+1)}$

**Proof.**

$$\begin{aligned} &\Pr[\text{Both parties accept} | i^* \text{ is defined}] \\ &\geq \Pr[i^* \in \mathcal{B} | i^* \text{ is defined}] - \Pr[\neg E | i^* \text{ is defined}] \\ &\geq \mu(\mathcal{G}) / 2^{20(I^B+1)} - \Pr[\neg E | i^* \text{ is defined}] && \text{(by Lemma 4.6)} \\ &\geq \frac{9}{10} \cdot 2^{-20(I^B+1)} - \frac{10}{1 - e^{-10}} \cdot 2^{-20(I^A+1)-20(I^B+1)} && \text{(by Lemma 4.5, Claim 4.2)} \\ &> \frac{8}{10} \cdot 2^{-20(I^B+1)}. \end{aligned}$$

where the last inequality follows from  $I^A \geq 0$ . ◀

► **Lemma 4.8.**  $\Pr[\text{Both parties accept}] \geq \frac{7}{10} \cdot 2^{-20(I^B+1)}$ .

**Proof.**

$$\begin{aligned} \Pr[\text{Both parties accept}] &= \Pr[i^* \text{ is defined}] \Pr[\text{Both parties accept} | i^* \text{ is defined}] \\ &\geq \frac{8(1 - e^{-10})}{10 \cdot 2^{20(I^B+1)}} > \frac{7}{10} \cdot 2^{-20(I^B+1)}, \end{aligned}$$

which follows from Lemma 4.7 and Lemma 4.3. ◀

► **Lemma 4.9.**  $|\pi(z_{i^*} = z | i^* \in \mathcal{B}) - \mu(z)| \leq 2(1 - \mu(\mathcal{G}))/\mu(\mathcal{G})$ .

**Proof.** By Lemma 4.4,

$$\Pr[i^* \in \mathcal{B} | i^* \text{ is defined}] \cdot \Pr[z_{i^*} = z | i^* \in \mathcal{B}] \leq \mu(z) \cdot 2^{-20(I^B+1)}.$$

Combining the above inequality and Lemma 4.6,

$$\Pr[z_{i^*} = z | i^* \in \mathcal{B}] \leq \mu(z)/\mu(\mathcal{G}).$$

For any set  $T \subseteq U$ ,

$$\sum_{z \in T} \pi(z_{i^*} = z | i^* \in \mathcal{B}) - \mu(z) \leq \sum_{z \in T} \mu(z)/\mu(\mathcal{G}) - \mu(z) \leq (1 - \mu(\mathcal{G}))/\mu(\mathcal{G}),$$

where the last inequality follows from  $\mu$  being a distribution. Therefore,

$$|\pi(z_{i^*} = z | i^* \in \mathcal{B}) - \mu(z)| = 2 \max_T \left( \sum_{z \in T} \pi(z_{i^*} = z | i^* \in \mathcal{B}) - \mu(z) \right) \leq 2(1 - \mu(\mathcal{G}))/\mu(\mathcal{G}).$$

◀

► **Lemma 4.10.**

$$\Pr[\neg E | i^* \in \mathcal{B}] < 0.01, \quad \Pr[\neg E | \text{Both parties accept}] < 0.01$$

**Proof.** We have,

$$\begin{aligned} \Pr[\neg E | i^* \in \mathcal{B}] &= \Pr[\neg E | i^* \in \mathcal{B}, i^* \text{ is defined}] \\ &= \frac{\Pr[\neg E \ \& \ i^* \in \mathcal{B} | i^* \text{ is defined}]}{\Pr[i^* \in \mathcal{B} | i^* \text{ is defined}]} \\ &\leq \frac{\Pr[\neg E | i^* \text{ is defined}]}{\Pr[i^* \in \mathcal{B} | i^* \text{ is defined}]} \\ &\leq \frac{10}{\mu(\mathcal{G})(1 - e^{-10})} \cdot 2^{-20(I^A+1)} \\ &< 0.01. \end{aligned}$$

The penultimate inequality follows from Lemmas 4.5, 4.6. Similarly,

$$\begin{aligned} \Pr[\neg E | \text{Both parties accept}] &= \Pr[\neg E | \text{Both parties accept}, i^* \text{ is defined}] \\ &= \frac{\Pr[\neg E \ \& \ \text{Both parties accept} | i^* \text{ is defined}]}{\Pr[\text{Both parties accept} | i^* \text{ is defined}]} \\ &\leq \frac{\Pr[\neg E | i^* \text{ is defined}]}{\Pr[\text{Both parties accept} | i^* \text{ is defined}]} \\ &\leq \frac{100}{8(1 - e^{-10})} \cdot 2^{-20(I^A+1)} \\ &< 0.01. \end{aligned}$$

The penultimate inequality follows from Lemmas 4.5, 4.7. ◀

► **Lemma 4.11.**  $|\mu(z) - \pi(z | \text{Both parties accept})| < 0.35$

**Proof.** Applying Proposition 2.2 twice gives,

$$\begin{aligned} & |\mu(z) - \pi(z|\text{Both parties accept})| \\ & \leq |\pi(z_{i^*} = z|i^* \in \mathcal{B}, E) - \mu(z)| + 4 \Pr[\neg E|\text{Both parties accept}] \\ & \leq |\pi(z_{i^*} = z|i^* \in \mathcal{B}) - \mu(z)| + 4 \Pr[\neg E|i^* \in \mathcal{B}] + 4 \Pr[\neg E|\text{Both parties accept}]. \end{aligned}$$

Applying Lemmas 4.9, 4.10 give,

$$|\mu(z) - \pi(z|\text{Both parties accept})| < 2(1 - \mu(\mathcal{G}))/\mu(\mathcal{G}) + 0.04 + 0.04 < 0.35,$$

as required.  $\blacktriangleleft$

In Theorem 4.1, property 2 is guaranteed by Lemma 4.8 and property 3 is guaranteed by Lemma 4.11.

### 4.1 Proof of Theorem 1.2

Define  $U$  to be the set of all transcripts of protocol  $\pi$ . For every  $m \in U$ , define  $\pi_x(m) = \pi(m|x)$ ,  $\pi_y(m) = \pi(m|y)$ ,  $\pi_{xy}(m) = \pi(m|xy)$ . We have,

$$I_\pi(X; M|YR) = \mathbb{E}_{xyr} \left[ \mathbb{D} \left( \frac{\pi_{xy}(m|r)}{\pi_y(m|r)} \right) \right].$$

Define  $\Gamma = \left\{ (x, y, r) \mid \mathbb{D} \left( \frac{\pi_{xy}(m|r)}{\pi_y(m|r)} \right) \leq 50 \cdot I_\pi^A, \mathbb{D} \left( \frac{\pi_{xy}(m|r)}{\pi_x(m|r)} \right) \leq 50 \cdot I_\pi^B \right\}$ . By a union bound and Markov's inequality,

$$\Pr[(x, y, r) \in \Gamma] \geq \frac{24}{25} \tag{6}$$

The following observations are useful.

$$\begin{aligned} \pi(m|xyr) &= \prod_{i:m_i \text{ sent by Alice}} \pi(m_i|xyrm_{<i}) \cdot \prod_{i:m_i \text{ sent by Bob}} \pi(m_i|xyrm_{<i}) \\ &= \prod_{i:m_i \text{ sent by Alice}} \pi(m_i|xrm_{<i}) \cdot \prod_{i:m_i \text{ sent by Bob}} \pi(m_i|yrm_{<i}), \end{aligned}$$

where the last inequality follows from the fact that Alice's messages depend only on  $x$ , the public randomness and the previous messages and Bob's messages depend only on  $y$ , public randomness and the previous messages. Similar expressions can be written for  $\pi(m|xr)$  and  $\pi(m|yr)$ .

We can now apply Theorem 4.1, with  $\mu(m) = \pi_{xy}(m|r)$ ,  $p(m) = \pi_x(m|r)$ ,  $q(m) = \pi_y(m|r)$ . The theorem implies that for every  $(x, y, r) \in \Gamma$ , there exists a constant  $c$  and a randomized protocol  $\tau$  with communication  $O(I_\pi^A)$  that samples a transcript  $m$  such that

$$|\pi(m) - \tau(m|\text{Both parties accept})| < 0.35, \Pr[\text{Both parties accept in } \tau] \geq 2^{-c(I_\pi^B+1)} \tag{7}$$

► **Lemma 4.12.**  $\mathbb{E}_{\pi(xy)} |\pi(m|xyr) - \tau(m|xyr, \text{Both parties accept})| < \frac{1}{25} + 0.35$

**Proof.**  $\tau$  guarantees that for  $(x, y, r) \in \Gamma$ ,  $|\pi(m|xyr) - \tau(m|xyr, \text{Both parties accept})| < 0.35$ . Therefore,

$$\begin{aligned} & \mathbb{E}_{\pi(xy)} |\pi(m|xyr) - \tau(m|xyr, \text{Both parties accept})| \\ & \leq \Pr[(x, y, r) \in \Gamma] \max_{(x,y,r) \in \Gamma} |\pi(z|xyr) - \tau(z|xyr, \text{Both parties accept})| + \Pr[(x, y, r) \notin \Gamma] \\ & < 0.35 + \frac{1}{25}, \end{aligned}$$

where the last inequality follows from (6).  $\blacktriangleleft$

**Input:**  $x, y$ , the inputs to  $\pi$ . A parameter  $t$ .  
**Public Randomness:** Sequence of strings  $L_1, \dots, L_t, r$ . A random transcript  $m'$ .  
 $i=1$ ;  
**while**  $i \leq t$  **do**  
    Run protocol  $\tau$  with  $L_i$  as the public random tape;  
    **if**  $\tau$  *Accepts* **then return** the output of  $\tau$ ;  
    **else**  $i=i+1$ ;  
**end**  
**return**  $m'$ ;

■ **Figure 3** Protocol  $\Sigma$  simulating  $\pi$ .

We now show a simulation of  $\pi$ . Run  $\Sigma$  shown in Figure 3 with parameter  $t = 10 \cdot 2^{c(I^B+1)}$ . We have,

$$CC(\Sigma) \leq t \cdot CC(\tau) = 10 \cdot I^A 2^{c(I^B+1)}.$$

Let  $J$  be the value of  $i$  at the time of termination of  $\Sigma$ .

► **Lemma 4.13.**  $\Pr[J = t + 1] \leq 1/25 + e^{-10}$

**Proof.** Conditioned on  $(x, y) \in \Gamma$ , the probability that  $\tau$  does not accept in iteration  $i$  equals  $1 - \Pr[\text{Both parties accepts}]$ . Therefore,

$$\begin{aligned} & \Pr[J = t + 1 | (x, y, r) \in \Gamma] \\ &= (1 - \Pr[\text{Both party accepts in } \tau])^t \\ &\leq \left(1 - 2^{-c(I^B+1)}\right)^t && \text{(by (7))} \\ &\leq e^{-10} && ((1-x)^n \leq e^{-xn}, x > 0). \end{aligned} \quad (8)$$

Now,

$$\begin{aligned} & \Pr[J = t + 1] \\ &= \Pr[(x, y, r) \in \Gamma] \cdot \Pr[J = t + 1 | (x, y, r) \in \Gamma] \\ &\quad + \Pr[(x, y, r) \notin \Gamma] \cdot \Pr[J = t + 1 | (x, y, r) \notin \Gamma] \\ &\leq e^{-10} + 1/25, \end{aligned}$$

where the last inequality follows from (6), (8). ◀

Let us now analyze the  $\ell_1$  distance between  $\Sigma$  and  $\pi$ .

► **Lemma 4.14.**  $|\Sigma - \pi| < 3/25 + 2e^{-10} + 0.35$

**Proof.** Conditioning on  $J \leq t$ , we know that  $\Sigma$ 's output corresponds to the output of  $\tau$ . Therefore,

$$|\pi(m) - \Sigma(m|J \leq t)| = \mathbb{E}_{\pi(xy_r)} |\pi(m|xy_r) - \tau(m|xy_r, \text{Both parties accept})|$$

By Proposition 2.1 and Lemma 4.13,

$$\begin{aligned} |\Sigma - \pi| &\leq 2 \Pr[J = t + 1] + |\pi(z) - \Sigma(z|J \leq t)| \\ &= 2 \Pr[J = t + 1] + \mathbb{E}_{\pi(xy_r)} |\pi(m|xy_r) - \tau(m|xy_r, \text{Both parties accept})| \\ &< 2/25 + 2e^{-10} + 1/25 + 0.35, \end{aligned}$$

where the last inequality follows from Lemma 4.13 and Lemma 4.12. ◀

This concludes the proof of Theorem 1.2.

## 4.2 A Rectangle Lower Bound

In this subsection, we show a corollary to Theorem 1.2. The simulation of Theorem 1.2 shows the existence of almost monochromatic rectangles of the right dimension.

► **Corollary 4.15.** *Given a protocol  $\pi$  with internal information  $(I_A, I_B)$  and inputs drawn from  $\mu$ , there exists a zero communication protocol for sampling a message  $m$  such that,*

- $\mu(\text{Alice Accepts}) = 2^{-O(I_A)}$
- $\mu(\text{Bob Accepts}|\text{Alice Accepts}) \geq 2^{-O(I_B)}$

*Moreover, given that both parties accept, the distribution of their samples is  $\delta$ -close in  $\ell_1$  distance to the distribution where both parties sample consistently from  $\pi(m)$ .*

**Proof.** First we wish to fix the public randomness of  $\pi$ , such that the internal information and the error bound are within limit. We have,

$$I_A = \mathbb{E}_r [I(M; X|Yr)], \quad I_B = \mathbb{E}_r [I(M; Y|Xr)], \quad \mathbb{E}_r [\mu(\pi(x, y) \neq f(x, y)|r)] \leq \epsilon$$

By Markov's inequality,

$$\begin{aligned} \Pr_r [I(M; X|Yr) > 3I_A] &< 1/3 \\ \Pr_r [I(M; Y|Xr) > 3I_B] &< 1/3 \\ \Pr_r [\mu(\pi(x, y) \neq f(x, y)|r) > 3\epsilon] &< 1/3. \end{aligned}$$

Therefore, by a union bound, there exists an  $r$  such that

$$I(M; X|Yr) \leq 3I_A, \quad I(M; Y|Xr) \leq 3I_B, \quad \mu(\pi(x, y) \neq f(x, y)|r) \leq 3\epsilon$$

After fixing  $r$ , we now have a protocol with internal information at most  $(3I_A, 3I_B)$  and error at most  $3\epsilon$  and no public randomness.

The following observations are useful.

$$\begin{aligned} \pi(m|xy) &= \prod_{i:m_i \text{ sent by Alice}} \pi(m_i|xym_{<i}) \cdot \prod_{i:m_i \text{ sent by Bob}} \pi(m_i|xym_{<i}) \\ &= \prod_{i:m_i \text{ sent by Alice}} \pi(m_i|x m_{<i}) \cdot \prod_{i:m_i \text{ sent by Bob}} \pi(m_i|y m_{<i}), \end{aligned}$$

where the last inequality follows from the fact that Alice's messages depend only on  $x$ , the public randomness and the previous messages and Bob's messages depend only on  $y$ , public randomness and the previous messages.

Define,

$$\pi_A(m|x) = \prod_{i:m_i \text{ sent by Alice}} \pi(m_i|x m_{<i}); \quad \pi_B(m|y) = \prod_{i:m_i \text{ sent by Bob}} \pi(m_i|y m_{<i})$$

In a similar fashion define,

$$\pi'_A(m|x) = \prod_{i:m_i \text{ sent by Bob}} \pi(m_i|y m_{<i}); \quad \pi'_B(m|y) = \prod_{i:m_i \text{ sent by Alice}} \pi(m_i|x m_{<i})$$

We are now in a position to describe the simulation. Consider the simulation  $\tau$  in Figure 4 (take  $c$  to be a large constant):

Let  $\mathcal{L}$  denote the sequence of  $L$  tuples in the public tape. Theorem 1.2 implies,



Simulation
<p><b>Public randomness:</b> A sequence of <math>L = 10 \cdot  U  \cdot 2^{c(I^A+1)}</math> tuples <math>(z_i, a_i, b_i) \in \mathcal{M} \times [0, 2^{c(I^A+1)}] \times [0, 2^{c(I^B+1)}]</math>, for <math>i = 1, 2, \dots, L</math>, <math>r</math> and a random function <math>h : [L] \rightarrow [2^{2c(I^A+1)}]</math>, where <math>\mathcal{M}</math> is the set of all transcripts of <math>\pi</math>.</p> <ol style="list-style-type: none"> <li>1. Alice computes the set <math>\mathcal{A} = \left\{ i \mid a_i \leq \pi_A(z_i xr), b_i \leq \pi'_A(z_i xr) \cdot 2^{c(I^B+1)} \right\}</math>, and Bob computes the set <math>\mathcal{B} = \left\{ i \mid a_i \leq \pi'_B(z_i yr) \cdot 2^{c(I^A+1)}, b_i \leq \pi_B(z_i yr) \right\}</math>.</li> <li>2. Alice computes <math>i^*</math>, the smallest element of <math>\mathcal{A}</math>.</li> <li>3. Alice accepts if <math>h(i^*) = 0^{2c(I^A+1)}</math>.</li> <li>4. If there is a unique <math>i \in \mathcal{B}</math> such that <math>h(i) = 0^{2c(I^A+1)}</math>, Bob accepts and assumes that the outcome of the protocol is <math>z_i</math>. Otherwise Bob aborts.</li> </ol>

■ **Figure 4** The sampling procedure.

- $\mathbb{E}_{\mathcal{L}, h} [\mu(\text{Alice Accepts})] \geq 2^{-O(I_A)}$  (this follows from  $h$  being a random hash function)
- $\mathbb{E}_{\mathcal{L}, h} [\mu(\text{Bob Accepts} | \text{Alice Accepts})] \geq 2^{-O(I_B)}$
- Given both parties accept, the distribution of the samples is  $\delta/3$ -close in  $\ell_1$  distance to the distribution where both parties sample the same from  $\pi(x, y)$ , for all constants  $\delta > 0$ . The third condition translates to,  $\mathbb{E}_{\mathcal{L}, h} [|\pi(m|xy) - \tau(m|xy, \text{Both parties accept})|] \leq \delta/3$ . This implies that there exists a fixing of  $\mathcal{L}, h$  such that

$$\mu(\text{Alice Accepts}) \geq 2^{-O(I_A)}, \quad \mu(\text{Bob Accepts} | \text{Alice Accepts}) \geq 2^{-O(I_B)}$$

$$|\pi(m|xy) - \tau(m|xy, \text{Both parties accept})| \leq \delta.$$

(the argument is similar to the one used for fixing of public randomness of  $\pi$ . One applies 3 Markov inequalities followed by an union bound) This completes the proof of the corollary. ◀

► **Corollary 4.16 (Restated).** *Given any randomized protocol  $\pi$  computing a boolean function  $f$ , with internal information  $(I_A, I_B)$ , there exists sets  $S, T$  and  $z \in \{0, 1\}$  such that*

$$\mu(x \in S) \geq 2^{-O(I_A)}, \quad \mu(y \in T | x \in S) \geq 2^{-O(I_B)}$$

$$\mu(f(x, y) \neq z | S \times T) \leq 3\epsilon + \delta,$$

where  $\epsilon$  is the error incurred by  $\pi$  under  $\mu$  and  $\delta > 0$  being any constant.

**Proof.** The protocol in Corollary 4.15 is deterministic. Any transcript in a deterministic protocol corresponds to a rectangle. Therefore, when both parties accepting, it exactly corresponds to a rectangle  $S \times T$  with

$$\mu(x \in S) \geq 2^{-O(I_A)}, \quad \mu(y \in T | x \in S) \geq 2^{-O(I_B)}.$$

In addition, conditioning on the event that both parties accept, the output  $z \in \{0, 1\}$  of the protocol (the value of the function that both parties agree on) has the property that  $\mu(f(x, y) \neq z) \leq 3\epsilon + \delta$ , where  $\epsilon$  is the error incurred by the the protocol  $\pi$  under  $\mu$ . ◀

### 4.2.1 Application – Lower Bounds for Lopsided Set Disjointness

In this subsection, we use the rectangle lower bound to reprove the well known lower bound for *lopsided* set disjointness by Pătraşcu in [18]. The problem of lopsided set disjointness is defined as follows,

► **Definition 4.17.** The set disjointness function on sets  $x, y \subseteq [NB]$  is

$$\text{SD}(x, y) = \begin{cases} 1 & \text{if } x \cap y = \emptyset, \\ 0 & \text{otherwise.} \end{cases}$$

*Lopsided set disjointness*(LSD) is a restricted version of the problem where we are promised that  $x, y \subseteq [NB]$  and  $|x| = N$ . The following bound proved by Patrascu [18] has found many applications to proving data structure lower bounds. The bound was shown to be tight by Saglam [20].

► **Theorem 4.18** ([18]). *For any protocol computing LSD with error probability  $\epsilon$ , one of the following holds,*

- *Alice communicates at least  $\gamma N \log B$  bits,*
- *Bob communicates at least  $NB^{1-c\gamma}$  bits,*

where  $c = c_1 + 1 + \frac{1}{\gamma \log B} (\log 2c_1 - \log(1 - h(12\epsilon + \delta)))$ , for a constant  $c_1$  and  $B = \Omega(1)$ .

Here we give a slightly different proof of Theorem 4.18, using Corollary 4.15. The universe is taken to be  $(2^{[B]})^N$ , the cartesian product of  $N$  power sets of  $[B]$ .  $(x, y) \in (2^{[B]}, 2^{[B]})$  is restricted tuples with  $|x| = 1$  and  $y$  takes exactly one element from the pair  $(2k, 2k + 1)$ , for all  $2k, 2k + 1 \in [B]$ . We define two distributions  $\psi$  and  $\mu$  on  $(x, y)$  as follows,

- $\psi$  is a uniform distribution on all such pairs  $(x, y)$ , with  $\text{LSD}(x, y) = 1$ .
- $\mu$  is a uniform distribution on all such pairs  $(x, y)$ .

The hard distribution for disjointness  $\mu_h$  is one where  $i \in [N]$  chosen at random, with  $(x_i, y_i)$  drawn from distribution  $\mu$  and rest of the coordinates  $(x_j, y_j)$ , for  $j \in [N] \setminus \{i\}$  is drawn i.i.d from  $\psi$ .

Having described the hard distribution, we are all set to describe the proof of Theorem 4.18.

**Proof.** We assume the contrary that there exists a protocol  $\pi$  computing  $\text{LSD}(x, y)$  on the distribution  $\mu_h$  with Alice communicating  $a < \gamma N \log B$  bits and Bob communicating  $b < NB^{1-c\gamma}$  bits.

We now use protocol  $\pi$  to compute LSD on a single block. Consider the case when the inputs  $(x, y)$  is drawn according to the distribution  $\psi$ . We know that protocol  $\pi$  computes LSD on inputs drawn i.i.d in  $\psi$  with information  $(I^A, I^B) < (\gamma N \log B, NB^{1-c\gamma})$ . By Theorem 2.11, there exist a protocol  $\tau$  computing  $\text{LSD}(x, y)$  on  $\psi$  with information  $(I_\tau^A, I_\tau^B) \leq (\gamma \log B, B^{1-c\gamma})$ .

Define  $\mathcal{M} = \{m | \mu(\text{LSD}(x, y) \neq \tau(x, y) | m) \leq 4\epsilon\}$ , a subset of all transcripts of  $\tau$ . Note that  $\mu(\mathcal{M}) \geq 1 - \frac{1}{4} = \frac{3}{4}$ , using the fact that  $\mu(\text{LSD}(x, y) \neq \tau(x, y)) \leq \epsilon$ . Therefore,  $\psi(\mathcal{M}) \geq 1 - \frac{2}{4} = \frac{1}{2}$ , since density of  $\text{Supp}(\psi)$  under  $\mu$  is one half.

First observe that Corollary 4.15 holds when the transcripts are restricted to the set  $\mathcal{M}$ . Now, Corollary 4.16 shows the existence of constant  $c_1$  and sets  $S, T$  such that  $\psi(x \in S) \geq 2^{-c_1(I^A)} \geq 2^{-c_1(\gamma \log B)}$  and  $\psi(y \in T | x \in S) \geq 2^{-c_1(I^B)} \geq 2^{-c_1(B^{1-4\gamma})}$ . We used with upper bounds on information  $(I^A, I^B)$  under  $\psi$ .

Since restricted to  $m \in \mathcal{M}$ ,  $\mu(\text{LSD}(x, y) \neq 1 | S \times T) \leq 3 \times 4\epsilon + \delta$ , where the factor 3 is an outcome of an averaging argument (see Corollary 4.16) and  $\delta$  corresponds to the  $\ell_1$  distance between the simulation in Corollary 4.15 and the actual protocol.

The marginal distribution in  $x$  being the same on  $\mu$  and  $\psi$  imply,

$$\mu(x \in S) \geq 2^{-c_1(\gamma \log B)}$$

Also,  $\mu(y) \geq \frac{1}{2} \cdot \psi(y)$  since  $\psi(x, y) = \mu(x, y|x \cap y = \emptyset)$  and  $\mu(x \cap y = \emptyset) = \frac{1}{2}$ . Therefore,

$$\mu(y \in T) \geq \frac{1}{2} \cdot \psi(y \in T) \geq 2^{-c_1(\gamma \log B + B^{1-c\gamma})-1}.$$

From here on, we work only with the distribution  $\mu$ . The bounds on probabilities imply the following bounds on the corresponding entropy,

$$\mathsf{H}(X|S) \geq (1 - c_1\gamma) \log B \quad (9)$$

$$\mathsf{H}(Y|T) \geq \frac{B}{2} - c_1\gamma \log B - c_1B^{1-c\gamma} - 1. \quad (10)$$

The error bound implies,

$$\forall x \in S, \mu(Y_x = 1|T) \leq \mu(\pi(x, y) \neq 1|S \times T) \leq 12\epsilon + \delta. \quad (11)$$

Note that  $Y_x$  is the projection of the vector  $Y$  (the indicator random variable for the subset in  $\{0, 1\}^B$ ) onto the coordinate indexed by  $x$ . This implies,  $\forall x \in S \mathsf{H}(Y_x|T) \leq \mathsf{h}(12\epsilon + \delta)$ .

Using subadditivity of entropy, we upper bound  $\mathsf{H}(Y|T)$  by  $\mathsf{H}(Y_S|T) + \mathsf{H}(Y_{\bar{S}}|T)$ , where  $\bar{S}$  is the complement of the set  $S$ .  $Y_S, Y_{\bar{S}}$  are projections of vector  $Y$  onto coordinates indexed by elements of sets  $S$  and  $\bar{S}$ . The first term in the expression can be simplified (using subadditivity of entropy) as follows,

$$\mathsf{H}(Y_S|T) \leq \mathsf{h}(12\epsilon + \delta) \cdot |S|$$

where the inequality follows from subadditivity of entropy and (11).

The second term yields,

$$\mathsf{H}(Y_{\bar{S}}|T) \leq \frac{B}{2} - |S|,$$

by an application of subadditivity of entropy and upper bounding binary entropy by 1.

(9) implies  $|S| \geq B^{1-c_1\gamma}$ . Therefore

$$\mathsf{H}(Y|T) \leq \frac{B}{2} - (1 - [\mathsf{h}(12\epsilon + \delta)]) B^{1-c_1\gamma}.$$

Equation (10) implies,

$$\mathsf{H}(Y|T) \geq \frac{B}{2} - c_1\gamma \log B - c_1B^{1-c\gamma} - 1$$

This yields a contradiction, as  $c > c_1 + \frac{1}{\gamma \log B} (\log 2c_1 - \log (1 - \mathsf{h}(12\epsilon + \delta)))$  and  $B = \Omega(1)$  imply,

$$\frac{B}{2} - (1 - [\mathsf{h}(12\epsilon + \delta)]) B^{1-c_1\gamma} < \frac{B}{2} - c_1\gamma \log B - c_1B^{1-c\gamma} - 1.$$

This concludes the proof of the theorem. ◀

**Acknowledgements.** We thank Mert Saglam and Makrand Sinha for useful discussions. We also thank the anonymous reviewer for pointing out an error in the previous version of Lemma 3.5.

---

References

---

- 1 Farid Ablayev. Lower bounds for one-way probabilistic communication complexity. In Andrzej Lingas, Rolf Karlsson, and Svante Carlsson, editors, *Proceedings of the 20th International Colloquium on Automata, Languages, and Programming*, volume 700 of *LNCS*, pages 241–252. Springer-Verlag, 1993.
- 2 Alexandr Andoni, Piotr Indyk, and Mihai Pătraşcu. On the optimality of the dimensionality reduction method. In *Proc. 47th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 449–458, 2006.
- 3 Reuven Bar-Yehuda, Benny Chor, Eyal Kushilevitz, and Alon Orlitsky. Privacy, additional information and communication. *IEEE Transactions on Information Theory*, 39(6):1930–1943, 1993. Preliminary version in CCC’90.
- 4 Boaz Barak, Mark Braverman, Xi Chen, and Anup Rao. How to compress interactive communication. In *Proceedings of the 2010 ACM International Symposium on Theory of Computing*, pages 67–76, 2010.
- 5 Allan Borodin, Rafail Ostrovsky, and Yuval Rabani. Lower bounds for high dimensional nearest neighbor search and related problems. In *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing (STOC’99)*, pages 312–321, New York, May 1999. Association for Computing Machinery.
- 6 Mark Braverman. Interactive information complexity. *Electronic Colloquium on Computational Complexity (ECCC)*, 18:123, 2011.
- 7 Mark Braverman. Interactive information complexity. In *Proceedings of the 44th symposium on Theory of Computing*, STOC’12, pages 505–524, New York, NY, USA, 2012. ACM.
- 8 Mark Braverman and Anup Rao. Information equals amortized communication. In Rafail Ostrovsky, editor, *FOCS*, pages 748–757. IEEE, 2011.
- 9 Mark Braverman and Omri Weinstein. A discrepancy lower bound for information complexity. *CoRR*, abs/1112.2000, 2011.
- 10 Amit Chakrabarti, Yaoyun Shi, Anthony Wirth, and Andrew Yao. Informational complexity and the direct sum problem for simultaneous message complexity. In *Proceedings of the 42nd Annual IEEE Symposium on Foundations of Computer Science*, pages 270–278, 2001.
- 11 Anat Ganor, Gillat Kol, and Ran Raz. Exponential separation of information and communication for boolean functions. *Electronic Colloquium on Computational Complexity (ECCC)*, 21:113, 2014.
- 12 T. S. Jayram, Subhash Khot, Ravi Kumar, and Yuval Rabani. Cell-probe lower bounds for the partial match problem. *J. Comput. Syst. Sci.*, 69(3):435–447, 2004.
- 13 Iordanis Kerenidis, Sophie Laplante, Virginie Lerais, Jérémie Roland, and David Xiao. Lower bounds on information complexity via zero-communication protocols and applications. *Electronic Colloquium on Computational Complexity (ECCC)*, 19:38, 2012.
- 14 Eyal Kushilevitz and Noam Nisan. *Communication complexity*. Cambridge University Press, Cambridge, 1997.
- 15 Peter Miltersen, Noam Nisan, Shmuel Safra, and Avi Wigderson. On data structures and asymmetric communication complexity. *Journal of Computer and System Sciences*, 57:37–49, 1 1998.
- 16 Peter Bro Miltersen. Lower bounds for union-split-find related problems on random access machines. In *Proceedings of the 26th Annual Symposium on the Theory of Computing*, pages 625–634, New York, May 1994. ACM Press.
- 17 Peter Bro Miltersen. Cell probe complexity – a survey. In V. Raman, C. Pandu Rangan, and R. Ramanujam, editors, *Proceedings of the 19th Conference on Foundations of Software Technology and Theoretical Computer Science, Chennai, India, December 13–15, 1999*, volume 1738 of *Lecture Notes in Computer Science*. Springer-Verlag, 1999.

- 18 Mihai Pătraşcu. Unifying the landscape of cell-probe lower bounds. *SIAM Journal on Computing*, 40(3):827–847, 2011. See also FOCS’08, arXiv:1010.3783.
- 19 Mihai Pătraşcu and Mikkel Thorup. Higher lower bounds for near-neighbor and further rich problems. *SIAM Journal on Computing*, 39(2):730–741, 2010. See also FOCS’06.
- 20 Mert Saglam. Private communication, 2014.
- 21 Michael Saks and Xiaodong Sun. Space lower bounds for distance approximation in the data stream model. In ACM, editor, *Proceedings of the 34th Annual ACM Symposium on Theory of Computing*, pages 360–369. ACM Press, 2002.

# Majority is Incompressible by $AC^0[p]$ Circuits\*

Igor Carboni Oliveira<sup>1</sup> and Rahul Santhanam<sup>2</sup>

1 Department of Computer Science, Columbia University, USA  
oliveira@cs.columbia.edu

2 Laboratory for Foundations of Computer Science, University of Edinburgh, UK  
rsanthan@inf.ed.ac.uk

---

## Abstract

We consider  $\mathcal{C}$ -compression games, a hybrid model between computational and communication complexity. A  $\mathcal{C}$ -compression game for a function  $f: \{0, 1\}^n \rightarrow \{0, 1\}$  is a two-party communication game, where the first party Alice knows the entire input  $x$  but is restricted to use strategies computed by  $\mathcal{C}$ -circuits, while the second party Bob initially has no information about the input, but is computationally unbounded. The parties implement an *interactive* communication protocol to decide the value of  $f(x)$ , and the *communication cost* of the protocol is the maximum number of bits sent by Alice as a function of  $n = |x|$ .

We show that any  $AC_d^0[p]$ -compression protocol to compute  $\text{Majority}_n$  requires communication  $n/(\log n)^{2d+O(1)}$ , where  $p$  is prime, and  $AC_d^0[p]$  denotes polynomial size unbounded fan-in depth- $d$  Boolean circuits extended with modulo  $p$  gates. This bound is essentially optimal, and settles a question of Chattopadhyay and Santhanam (2012). This result has a number of consequences, and yields a tight lower bound on the total fan-in of oracle gates in constant-depth oracle circuits computing  $\text{Majority}_n$ .

We define *multiparty* compression games, where Alice interacts in parallel with a polynomial number of players that are not allowed to communicate with each other, and communication cost is defined as the sum of the lengths of the longest messages sent by Alice during each round. In this setting, we prove that the randomized  $r$ -round  $AC^0[p]$ -compression cost of  $\text{Majority}_n$  is  $n^{\Theta(1/r)}$ . This result implies almost tight lower bounds on the maximum individual fan-in of oracle gates in certain restricted bounded-depth oracle circuits computing  $\text{Majority}_n$ . Stronger lower bounds for functions in NP would separate NP from NC<sup>1</sup>.

Finally, we consider the round separation question for two-party  $AC^0$ -compression games, and significantly improve known separations between  $r$ -round and  $(r + 1)$ -round protocols, for any constant  $r$ .

**1998 ACM Subject Classification** F. Theory of Computation

**Keywords and phrases** interactive communication, compression, circuit lower bound

**Digital Object Identifier** 10.4230/LIPIcs.CCC.2015.124

## 1 Introduction

### 1.1 Motivation and Background

Computational complexity theory investigates the complexity of solving explicit problems in various computational models. While fairly strong lower bounds are known for restricted models such as constant-depth circuits (Ajtai [2], Furst, Saxe, and Sipser [23], Yao [49], and

---

\* This work was supported by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013)/ERC grant agreement no. 615075. The first author was also supported by NSF grants CCF-1116702 and CCF-1115703.

Håstad [27]) and monotone circuits (Razborov [35], Andreev [7], and Alon and Boppana [5]), our understanding of general Boolean circuits is still very limited. For example, our current state of knowledge does not rule out that every function in  $\text{NTIME}(2^n)$  is computed by Boolean circuits of linear size.

Several barriers have been identified to proving lower bounds for general Boolean circuits, such as relativization (Baker, Gill, and Solovay [9]), algebrization (Aaronson and Wigderson [1]), and the “natural proofs” barrier (Razborov and Rudich [37]). Most known lower bound techniques for restricted models are “naturalizable”, and it is believed that substantially different methods will be required in order to prove strong lower bounds for unrestricted models.

In spite of this, the techniques used to prove lower bounds for weaker models are still interesting, and an improved understanding of these techniques can have substantial benefits. First, there is a developing theory of connections between unconditional lower bounds and algorithmic results, which involves satisfiability algorithms, learning algorithms, truth-table generation, among other models (cf. Williams [46], Oliveira [33], and Santhanam [39]). In particular, such connections provide new insights and results in both areas, and a better understanding of restricted classes of circuits can lead to improved algorithms (cf. Williams [47]). Second, strong enough lower bounds for weaker models imply lower bounds for more general models (Valiant [43, 44], see Viola [45] for a modern exposition). In a similar vein, we mention the surprising results from Allender and Koucký [4] showing that, in some cases, weak circuit size lower bounds of the form  $n^{1+\epsilon}$  yield much stronger results.

Furthermore, even if known proof techniques individually naturalize, it is possible they could be used as ingredients of a more sophisticated approach which is more powerful. A recent striking example of this is the use by Williams [48] of structural characterizations of  $\text{ACC}^0$  circuits, together with various complexity tools such as completeness for problems on succinctly represented inputs, diagonalization, and the easy witness method, in order to separate  $\text{NEXP}$  from  $\text{ACC}^0$ . Given the paucity of techniques in the area of complexity lower bounds, it makes sense to try to properly understand the techniques we do have.

We focus in this work on  $\mathcal{C}$ -compression games (Chattopadhyay and Santhanam [13]), where  $\mathcal{C}$  is some class of Boolean circuits. A  $\mathcal{C}$ -compression game is a 2-player (interactive) communication game where the first player Alice is computationally bounded (by being restricted to play strategies in  $\mathcal{C}$ ) and has access to the entire input  $x \in \{0, 1\}^n$ , while the second player Bob is computationally unbounded and initially has no information about the input. Alice and Bob communicate to compute  $f(x)$  for a fixed Boolean function  $f: \{0, 1\}^n \rightarrow \{0, 1\}$ , and the question is how many bits of communication sent by Alice are required. Note that if  $f$  is computable by  $\mathcal{C}$ , then 1 bit of communication suffices, as Alice can compute  $f(x)$  by herself, and send the answer to Bob. Thus, if we are interested in unconditional lower bounds on the communication cost for an explicit function, we must study circuit classes  $\mathcal{C}$  where lower bounds are already known for explicit functions, such as constant-depth circuits, and their extension with modulo  $p$  gates.

Compression games hybridize between communication complexity and computational complexity as follows. In the traditional two-party communication complexity model, Alice and Bob are symmetric – they each know half of the input, and communicate to compute a given function on the whole input. Neither party is computationally bounded. Thus they are equally constrained (or unconstrained) informationally as well as computationally. In the compression game setting, an asymmetry appears. Alice now has an informational advantage over Bob – she begins with knowledge of the whole input, while Bob has no knowledge about the input at all. However, this informational advantage is offset by a computational

constraint – Alice can only use strategies computable from  $\mathcal{C}$ . Thus studying compression games can be thought of as studying the tradeoff between information and computation. Typically, when studying the question of lower bounds against  $\mathcal{C}$ , we are merely interested in whether a function  $f$  is computable in  $\mathcal{C}$  or not. Now, we are concerned instead by how much information can be obtained about  $f(x)$  using merely circuits from  $\mathcal{C}$ , or conversely, how much assistance a  $\mathcal{C}$ -bounded party requires from an unbounded one in order to compute  $f(x)$ . In other terms, we would like to obtain a refined quantitative picture of solvability by  $\mathcal{C}$ -circuits, rather than a purely qualitative one.

Communication complexity has long been an important tool in the complexity theorist's toolkit. In particular, several lower bound techniques such as the crossing sequence method, the Nečiporuk method [32] and the Khrapchenko method [28] can be interpreted as uses of communication complexity (cf. Kushilevitz and Nisan [30]). Often, when a computational model is relatively weak, lower bound techniques exploit some sort of information bottleneck in the model, which is how communication complexity enters the picture. By studying compression games, where the model explicitly incorporates both communication and computation, we hope to better understand the interplay between communication complexity techniques and computational complexity techniques.

We explore in this work the power of the *polynomial approximation method* (Razborov [36], Smolensky [40]) and the *random restriction method* (cf. Furst, Saxe, and Sipser [23] and Håstad [27]) in the context of interactive compression games. We use these techniques and the compression framework to prove significant generalizations of known lower bounds for constant-depth circuits.

Compression games have been considered before, both to prove unconditional and conditional lower bounds. The pioneering work of Dubrov and Ishai [17] showed that  $\text{Parity}_n$  requires  $\text{AC}^0$ -compression cost  $n^{1-\varepsilon}$  (for any fixed  $\varepsilon > 0$ , and large enough  $n$ ) when there is only one round of communication between Alice and Bob. Dubrov and Ishai were motivated by questions about the randomness complexity of sampling, and their work has later found applications in leakage-resilient cryptography (Faust et al. [18]). Chattopadhyay and Santhanam [13] strengthened the Dubrov-Ishai lower bound to  $n/\text{poly}(\log n)$ , and showed that the lower bound holds for multi-round games where Alice is allowed to use a randomized strategy. Their main technique was a generic connection between correlation and multi-round compression. As strong correlation lower bounds are not known for  $\text{AC}^0[p]$  circuits (see e.g. Srinivasan [41]), their technique does not yield strong lower bounds for multi-round  $\text{AC}^0[p]$ -compression games, which constitute the main topic of this work.

The investigation of single-round compression (also known as instance compression) has found connections to other topics in areas such as cryptography (Harnik and Naor [26]), parameterized complexity (cf. Bodlaender et al. [10]), probabilistic checkable proofs (Fortnow and Santhanam [22]), and structural complexity (Buhrman and Hitchcock [12]), and has received considerable attention recently (see e.g. Drucker [16] and Dell [14]). There has also been a long line of work on proving lower bounds for  $\text{SIZE}(\text{poly}(n))$ -compression games under complexity-theoretic assumptions (cf. Dell and van Melkebeek [15]), but papers along this line use very different ideas, and hence are tangential to our work.

## 1.2 Main Results and Techniques

For a circuit class  $\mathcal{C}$ , we use  $\mathcal{C}_d$  to denote the restriction of  $\mathcal{C}$  to polynomial size circuits of depth  $d$ . For instance,  $\text{AC}_d^0$  refers to polynomial size depth- $d$  circuits. Recall that  $\text{Majority}_n: \{0, 1\}^n \rightarrow \{0, 1\}$  is the function that is 1 on an input  $x$  if and only if  $\sum_{i \in [n]} x_i \geq n/2$ . Further, we let  $\text{MOD}_q^n: \{0, 1\}^n \rightarrow \{0, 1\}$  be the function that is 1 if and only if  $q$  divides  $\sum_{i \in [n]} x_i$ .



The proof that  $\text{Majority}_n \notin \text{AC}_d^0[p]$  for  $d(n) = o(\log n / \log \log n)$  (Razborov [36], Smolensky [40]) remains one of the strongest lower bounds for an explicit function. There are no known explicit lower bounds for polynomial size circuits of depth  $d = \omega(\log n / \log \log n)$ , nor for constant depth circuits with arbitrary (composite) modulo gates.

In the framework of compression games, the Razborov-Smolensky lower bound is equivalent to the claim that in any  $\text{AC}^0[p]$  game for Majority, there must be non-trivial communication between Alice and Bob. More recently, Chattopadhyay and Santhanam [13] proved that in any *randomized single-round*  $\text{AC}_d^0[p]$ -compression protocol for this function, Alice must communicate  $\sqrt{n}/(\log n)^{O(d)}$  bits. However, their technique does not extend to multiple-round compression games. Before this work, the only known technique to prove unconditional lower bounds for games with an arbitrary number of rounds used a connection between compressibility and correlation. The lack of strong correlation bounds for low-degree  $\mathbb{F}_p$  polynomials computing explicit Boolean functions prevents us from using this connection to get  $\text{AC}^0[p]$ -compression lower bounds (see Srinivasan [41] for more details).

In this work, we bypass this difficulty through a new application of the polynomial approximation method, obtaining the following result.

► **Theorem 1.1.** *Let  $p$  be a prime number. There exists a constant  $c \in \mathbb{N}$  such that, for any  $d \in \mathbb{N}$ , and every  $n \in \mathbb{N}$  sufficiently large, the following holds.*

- (i) *Any  $\text{AC}_d^0[p]$ -compression game for Majority $_n$  (with any number of rounds) has communication cost at least  $n/(\log n)^{2d+c}$ .*
- (ii) *There exists a single-round  $\text{AC}_d^0$ -compression game for Majority $_n$  with communication cost at most  $n/(\log n)^{d-c}$ .*

The argument for the lower bound part of this result proceeds roughly as follows. First, we show via a reduction in the interactive compression framework that a protocol for Majority $_n$  can be used to compress other symmetric functions, such as  $\text{MOD}_q^n$ . In other words, it is enough to prove a strong communication lower bound for  $\text{MOD}_q^n$  in order to establish the lower bound in Theorem 1.1. We then employ a general technique that allows us to transform an interactive protocol for a Boolean function  $f$  into an exponentially large circuit computing  $f$ , following an approach introduced in Chattopadhyay and Santhanam [13]. We have thus reduced the original problem involving computation and communication to a certain circuit lower bound for  $\text{MOD}_q$ .

A crucial ingredient in our proof is a new exponential lower bound for a certain class of bounded-depth circuits extended with modulo  $p$  gates computing the  $\text{MOD}_q$  function. Although obtaining circuit lower bounds for depth  $d$  circuits beyond size roughly  $2^{n^{1/(d-1)}}$  is a major open problem in circuit complexity (see e.g. Viola [45]), we show that, under a certain *semantic* constraint on the  $\text{AC}_d^0[p]$  circuit,  $\text{MOD}_q^n$  requires circuits of size  $2^{n/(\log n)^{O(d)}}$ . More specifically, we consider circuits consisting of a disjunction of exponentially many polynomial size circuits, for which the following holds: whenever the top gate evaluates to true, precisely one subcircuit evaluates to true.

The proof of this circuit lower bound relies on the application of the polynomial approximation method in the *exponentially small error regime*, as opposed to the original proofs of Razborov and Smolensky, which are optimized with constant error. In particular, this approach allows us to prove a stronger lower bound that avoids the correlation barrier mentioned before. In order to implement this idea, we rely on a recent strengthening of their method introduced by Kopparty and Srinivasan [29], and on an extension of the degree lower bounds of Razborov and Smolensky to very small error. We believe that this new circuit lower bound may be of independent interest, and that semantic restrictions will find

more applications in circuit complexity. Altogether, these results give the lower bound in Theorem 1.1.

Theorem 1.1 implies a new result for  $\text{AC}^0[p]$  circuits extended with arbitrary oracle gates, which we state next.

► **Corollary 1.2.** *Let  $p \geq 2$  be prime, and  $d \in \mathbb{N}$ . There exists a constant  $c \in \mathbb{N}$  such that, for every sufficiently large  $n$ , the following holds. If  $\text{Majority}_n$  is computed by polynomial-size  $\text{AC}_d^0[p]$  circuits with arbitrary oracle gates, then the total fan-in of the oracle gates is at least  $n/(\log n)^{2d+c}$ .*

Another interesting consequence of Theorem 1.1 is that it provides information about the *structure* of polynomial size circuits with modulo  $p$  gates computing  $\text{Majority}_n$ . More precisely, it implies that in any layered circuit, at least  $\lfloor n/(\log n)^{2k+c} \rfloor$  gates must be present in the  $k$ -th layer, which is essentially optimal.

Observe that Theorem 1.1 holds for deterministic compression games. For randomized protocols, in which Alice can employ a probabilistic strategy, we use our techniques to prove the following strengthening over previous results.

► **Theorem 1.3.** *Let  $p$  and  $q$  be distinct primes. There exists a constant  $c \in \mathbb{N}$  such that, for any  $d \in \mathbb{N}$ , and  $n \in \mathbb{N}$  sufficiently large, every randomized  $\text{AC}_d^0[p]$ -compression game for  $\text{MOD}_q^n$  with any number of rounds and error at most  $1/3$  has communication cost at least  $\sqrt{n}/(\log n)^{d+c}$ .*

We stress that Theorems 1.1 and 1.3 hold both for  $\text{Majority}$  and  $\text{MOD}_q$ , whenever  $p \neq q$  are distinct primes. Determining the correct communication cost for probabilistic and average-case games for these functions remains an interesting open problem. (We discuss these models in more detail in Section 2.)

We also consider a model of *multiparty* compression games. In this framework, Alice is allowed to interact during each round with  $k$  additional parties, and the communication cost of the round is defined to be the length of the longest message sent by Alice to one of the parties. Further, the cost of the protocol on a given input is defined as the sum of the costs of the individual rounds. We stress that the extra parties are not allowed to interact with each other during the execution of the protocol.

This is a natural communication framework, motivated by the question of lower bounds for oracle circuits. Lower bounds in this model with a bounded number of rounds imply lower bounds on the maximum individual fan-in of oracle gates in oracle circuits with a bounded number of such layers.

We prove the following bounds on the randomized multiparty  $\text{AC}^0[p]$ -compression cost of  $\text{Majority}$ .

► **Theorem 1.4.** *Let  $p \in \mathbb{N}$  be a fixed prime. For every  $k, r, d \in \mathbb{N}$ , the following holds.*

- (i) *There exists a deterministic  $n^{1/r}$ -party  $r$ -round  $\text{AC}^0[p]$ -compression game for  $\text{Majority}_n$  with cost  $\tilde{O}(n^{1/r})$ .*
- (ii) *Every randomized  $n^k$ -party  $r$ -round  $\text{AC}_d^0[p]$ -compression game for  $\text{Majority}_n$  has cost  $\tilde{\Omega}(n^{1/2r})$ .*

The proof of Theorem 1.4 also employs the polynomial approximation method, although the argument is different in this case. Observe that this result says that the communication cost of  $\text{Majority}_n$  in the randomized multiparty framework is  $n^{\Theta(1/r)}$  for  $r$ -round protocols. In other words, allowing Alice to interact with more parties for more time reduces communication considerably (under the definition of communication cost for multiparty games).

We obtain a consequence of Theorem 1.4 for oracle circuits where there are a bounded number  $r$  of such layers, i.e., there are no more than  $r$  oracle gates on any input-output path in the circuit.

► **Corollary 1.5.** *Let  $p \geq 2$  be prime, and  $r, d \in \mathbb{N}$ . If  $\text{Majority}_n$  is computed by an  $\text{AC}_d^0[p]$  circuit of polynomial size with arbitrary oracle gates that contains at most  $r$  layers of such gates, then there is some oracle gate with fan-in at least  $\tilde{\Omega}(n^{1/2r})$ .*

In fact, lower bounds for multiparty games are connected to the NP versus  $\text{NC}^1$  question. It is possible to show that every Boolean function in  $\text{NC}^1/\text{poly}$  admits  $\text{poly}(n)$ -party  $r$ -round  $\text{AC}^0$ -compression games with cost  $n^{O(1/r)}$ . Thus, proving a lower bound of  $n^{\Omega(1)}$  on the cost of  $\text{poly}(n)$ -party  $\text{AC}^0$ -compression games with  $\omega(1)$  rounds for a function in NP would separate NP from  $\text{NC}^1/\text{poly}$ . We conjecture that such a lower bound holds for the Clique function. Note that it is already known that strong enough lower bounds on the size of constant-depth circuits for NP functions implies a separation between NP and  $\text{NC}^1$  (cf. Viola [45]). The novelty here is that sufficiently strong results about *polynomial-size* constant depth circuits imply similar separations. Essentially, the computation of logarithmic-depth circuits can be factored into constant-depth and low-communication components, and our multiparty communication game models precisely this mixture of notions.

There is an interesting contrast in the statement of Theorem 1.1: while the lower bound holds for protocols with any number of rounds, the upper bound is given by a single-round protocol. It is natural to wonder whether in the compression setting interaction allows Alice to solve more computational problems. We provide a natural example of the power of interaction in our framework in Section 6, where we observe that, while the inner product function cannot be computed by polynomial size  $\text{MAJ} \circ \text{MAJ}$  circuits (Hajnal et al. [25]), there exists an efficient two-party ( $\text{MAJ} \circ \text{MAJ}$ )-compression game for this function.

In a similar direction, a *quantitative* study of the power of interaction in two-party compression games was initiated by Chattopadhyay and Santhanam [13] (with respect to  $\text{AC}^0$ -compression games). They obtained a quadratic gap in communication when one considers  $r$  and  $(r - 1)$ -round protocols for a specific Boolean function. We obtain the following strengthening of their round separation theorem.

► **Theorem 1.6.** *Let  $r \geq 2$  and  $\varepsilon > 0$  be fixed parameters. There is an explicit family of functions  $f = \{f_n\}_{n \in \mathbb{N}}$  with the following properties:*

- (i) *There exists an  $\text{AC}_2^0(n)$ -bounded protocol  $\Pi_n$  for  $f_n$  with  $r$  rounds and cost  $c(n) \leq n^\varepsilon$ , for every  $n \geq n_f$ , where  $n_f$  is a fixed constant that depends on  $f$ .*
- (ii) *Any  $\text{AC}^0(\text{poly}(n))$ -bounded protocol  $\Pi$  for  $f$  with  $r - 1$  rounds has cost  $c(n) \geq n^{1-\varepsilon}$ , for every  $n \geq n_\Pi$ , where  $n_\Pi$  is a fixed constant that depends on  $\Pi$ .*

Our hard function is based on a pointer jumping problem with a grid structure, while Chattopadhyay and Santhanam uses a tree structure. Similar constructions have been used in other works in communication complexity in the information theoretic setting (Papadimitriou and Sipser [34], and subsequent works), but our analysis needs to take into account computational considerations as well.

The proof of Theorem 1.6 relies on a careful application of the *random restriction method*, coupled with a round elimination strategy. Observe that the upper bound is achieved by protocols where Alice's strategy can be implemented by linear-size DNFs, while the communication lower bound holds for polynomial size circuits.

### 1.3 Organization

We define interactive compression games and introduce notation in the next section. In Section 3, we give the proof of our main result, deferring the discussion of some auxiliary results to the Appendix. Multiparty compression games are discussed in Section 4, followed by applications of our communication lower bounds to circuits with oracle gates in Section 5. A natural example for which interactive compression can be easier than computation is presented in Section 6. The round separation theorem for  $\text{AC}^0$  games is proved in Section 7. Finally, we mention a few open problems and research directions in Section 8.

## 2 Preliminaries and Notation

The results of this paper are essentially self-contained, but some familiarity with basic notions from complexity theory and communication complexity can be helpful. A good introduction to these areas can be found in [8] and [30], respectively.

**Basic definitions.** For any positive integer  $m \in \mathbb{N}$ , let  $[m] \stackrel{\text{def}}{=} \{1, \dots, m\}$ . We use  $\text{Majority}_n$  to denote the Boolean function over  $n$  variables that is 1 if and only if  $\sum_i x_i \geq n/2$ . For a prime  $p$ , we let  $\text{MOD}_p^n$  be the Boolean function over  $n$  variables that is 1 if and only if  $p$  divides  $\sum_i x_i$ . We let  $\text{Parity}_n \stackrel{\text{def}}{=} \neg \text{MOD}_2^n$ . A function  $h: \{0, 1\}^n \rightarrow \{0, 1\}$  is *symmetric* if there exists a function  $\phi: [n] \rightarrow \{0, 1\}$  such that  $h(x) = \phi(\sum_i x_i)$ , for every  $x \in \{0, 1\}^n$ . Clearly,  $\text{Majority}_n$  and  $\text{MOD}_p^n$  are symmetric functions. We say that a Boolean function  $f$   $\varepsilon$ -approximates a Boolean function  $g$  over a distribution  $\mathcal{D}$  if  $\Pr_{\mathbf{x} \sim \mathcal{D}}[f(\mathbf{x}) \neq g(\mathbf{x})] \leq \varepsilon$ . An  $\varepsilon$ -error *probabilistic* polynomial  $\mathbf{Q}(x_1, \dots, x_n) \in \mathbb{F}_p[x_1, \dots, x_n]$  for a Boolean function  $f: \{0, 1\}^n \rightarrow \{0, 1\}$  is a distribution  $\mathcal{E}$  over polynomials such that, for every  $x \in \{0, 1\}^n$ ,  $\Pr_{\mathbf{Q} \sim \mathcal{E}}[f(x) \neq \mathbf{Q}(x)] \leq \varepsilon$ .<sup>1</sup> The degree of a probabilistic polynomial is the maximum degree over the polynomials on which  $\mathcal{E}$  is supported. We say that functions  $f: \{0, 1\}^n \rightarrow \{0, 1\}$  and  $g: \{0, 1\}^n \rightarrow \{0, 1\}$  are *disjoint* if  $f^{-1}(1) \cap g^{-1}(1) = \emptyset$ . Given a string  $w$ , we use  $|w|$  to denote the length of  $w$ , and  $|w|_1$  to denote the number of 1s in  $w$ . We will use  $p$  and  $q$  to denote prime numbers throughout the text, unless noted otherwise.

**Languages and circuit classes.** Given a language  $L \subseteq \{0, 1\}^*$ , we let  $L_n \stackrel{\text{def}}{=} L \cap \{0, 1\}^n$ . We view  $L_n$  as a Boolean function in the natural way. We will use  $\mathcal{C}$  to denote a circuit class, such as  $\text{AC}^0$  and  $\text{AC}^0[p]$ . Unless stated otherwise, assume that any circuit class discussed in this paper contains AND, OR, and NOT gates of unbounded fan-in. Our results hold with more general circuit classes, but we stick with this definition for simplicity. The size of a circuit corresponds to the total number of gates in the circuit. We use  $\mathcal{C}_d(s(n))$  to denote the same class restricted to circuits of depth  $d$  and size  $O(s(n))$ . For instance, we abuse notation and write  $\text{AC}_d^0[p](\text{poly}(n))$  to denote the set of languages decided by polynomial size circuits of depth at most  $d$  consisting of unbounded fan-in AND, OR, NOT and  $\text{MOD}_p$  gates, for a fixed prime  $p \in \mathbb{N}$ . As a convention, if we write  $\mathcal{C}$  without a depth and size specialization, assume that it consists of constant depth polynomial size circuits with gates from  $\mathcal{C}$ . As usual, we will identify  $\mathcal{C}$  both as a set of languages, and as a class of circuits, depending on the context. Furthermore, if  $C$  is a fixed circuit, we may also use  $C$  to refer to the Boolean function computed by this circuit. The correct meaning will always be clear in both cases.

<sup>1</sup> We will use boldface notation whenever we want to emphasize that we are referring to a random variable or a probability distribution over the corresponding structures.

**Deterministic compression games.** Given a circuit class  $\mathcal{C}$  and a language  $L$ , we define a communication game between two players Alice and Bob. The goal is to decide whether a given string  $x \in \{0, 1\}^n$  belongs to  $L$ . We describe this game informally as follows. Alice knows  $x$ , but her computational power is limited to functions computed by circuits from  $\mathcal{C}$ . On the other hand, Bob can perform arbitrary computations, but has no information about  $x$  during the beginning of the game. The players exchange messages during the execution of the protocol, and at the end should be able to decide whether  $x \in L$ . The goal is to compute the initial function correctly while minimizing the total number of bits sent by Alice during the game.

Formally, a  $\mathcal{C}$ -bounded protocol  $\Pi_n = \langle C^{(1)}, \dots, C^{(r)}, f^{(1)}, \dots, f^{(r-1)}, E_n \rangle$  with  $r = r(n)$  rounds consists of a sequence of  $\mathcal{C}$ -circuits for Alice, a strategy for Bob, given by functions  $f^{(1)}, \dots, f^{(r-1)}$ , and a set of accepting transcripts  $E_n$ . We associate to every protocol  $\Pi_n$  its signature  $\text{signature}(\Pi_n) = (n, s_1, t_1, \dots, t_{r-1}, s_r)$ , which is the sequence corresponding to the input size  $n = |x|$  and the length of the messages exchanged by Alice and Bob during the execution of the protocol. For convenience, let  $s = \sum_{i \in [r]} s_i$ , and  $t = \sum_{i \in [r-1]} t_i$ . We always have  $E_n \subseteq \{0, 1\}^{t+s}$ . In addition, we let  $\text{rounds}(\Pi_n) \stackrel{\text{def}}{=} r$ . For every  $i \in [r]$ ,

$$C^{(i)}: \{0, 1\}^{n + \sum_{j < i} (s_j + t_j)} \rightarrow \{0, 1\}^{s_i},$$

and for every  $i \in [r-1]$ ,

$$f^{(i)}: \{0, 1\}^{\sum_{j \leq i} s_j} \rightarrow \{0, 1\}^{t_i}.$$

In other words, before the beginning of the  $i$ -th round, Alice has sent messages  $a^{(1)}, \dots, a^{(i-1)}$  of size  $s_1, \dots, s_{i-1}$ , respectively, and Bob has replied with messages  $b^{(1)}, \dots, b^{(i-1)}$  of size  $t_1, \dots, t_{i-1}$ , respectively. The next message sent by Alice is given by

$$a^{(i)} \stackrel{\text{def}}{=} C^{(i)}(x, a^{(1)}, b^{(1)}, \dots, a^{(i-1)}, b^{(i-1)}).$$

On the other hand, since Bob has unlimited computational power, its message during the  $i$ -th round is given simply by  $b^{(i)} \stackrel{\text{def}}{=} f^{(i)}(a^{(1)}, \dots, a^{(i)})$ . The transcript of  $\Pi_n$  on  $x \in \{0, 1\}^n$  is the sequence of messages exchanged by Alice and Bob during the execution of the protocol on  $x$ , and will be denoted by  $\text{transcript}_{\Pi_n}(x) \stackrel{\text{def}}{=} \langle a^{(1)}, b^{(1)}, \dots, a^{(r)} \rangle \in \{0, 1\}^{s+t}$ . We say that  $\Pi_n$  solves the compression game of a function  $h_n: \{0, 1\}^n \rightarrow \{0, 1\}$  if

$$h(x) = 1 \iff \text{transcript}_{\Pi_n}(x) \in E_n.$$

Finally, we let  $\text{cost}(\Pi_n) \stackrel{\text{def}}{=} s$ . We stress that the length of the messages sent by Bob does not contribute to the cost of the protocol, and we assume for convenience that the length of these messages are limited by the size of the circuits in  $\mathcal{C}$ . Observe that a *single-round* game consists of a protocol  $\Pi_n$  with  $\text{signature}(\Pi_n) = (n, s_1)$ . Put another way, Alice sends a single message  $a^{(1)} \in \{0, 1\}^{s_1}$ , and a decision is made.

Given a language  $L$  and a circuit class  $\mathcal{C}$ , we say that a sequence of  $\mathcal{C}$ -bounded protocols  $\Pi = \{\Pi_n\}_{n \in \mathbb{N}}$  solves the compression game of  $L$  with cost  $c(n)$  and  $r(n)$  rounds if, for every  $n$ ,  $\Pi_n$  solves the compression game of  $L_n$ , and in addition satisfies  $\text{cost}(\Pi_n) \leq c(n)$  and  $\text{rounds}(\Pi_n) \leq r(n)$ .

Observe that if  $L \in \mathcal{C}$  then Alice can compute  $L(x)$  by herself, and there is a trivial protocol of cost  $c(n) = 1$  for  $L$ . On the other hand, for every language  $L$  there exists a protocol solving its compression game with cost  $c(n) \leq n$ , since Alice can simply send her whole input to Bob.

**Probabilistic and average-case compression games.** The definition presented before captures deterministic games computing a function correctly on every input  $x$ . Our framework can be extended naturally to probabilistic and average-case games.

First, in a *probabilistic*  $\mathcal{C}$ -compression game, Alice is allowed to use randomness when computing her next message, while Bob's strategy remains deterministic. Formally, each circuit  $C^{(i)}$  has an additional input of uniformly distributed bits, and different circuits have access to independent bits. Clearly, on any  $x \in \{0, 1\}^n$ ,  $\text{Transcript}_{\Pi_n}(x)$  is now a random variable distributed over  $\{0, 1\}^{s+t}$ . The other definitions remain the same. We say that  $\Pi_n$  solves the compression game of a function  $h_n: \{0, 1\}^n \rightarrow \{0, 1\}$  with error probability at most  $\gamma(n) \in [0, 1]$  if, for every  $x \in \{0, 1\}^n$ ,

$$\begin{aligned} h_n(x) = 1 &\implies \Pr_{\Pi_n}[\text{Transcript}_{\Pi_n}(x) \in E_n] \geq 1 - \gamma(n), \text{ and if} \\ h_n(x) = 0 &\implies \Pr_{\Pi_n}[\text{Transcript}_{\Pi_n}(x) \in E_n] \leq \gamma(n). \end{aligned}$$

On the other hand, in a *average-case*  $\mathcal{C}$ -compression game, we have deterministic games as defined before, but allow a small error during the computation of  $h_n$  with respect to the uniform distribution over  $\{0, 1\}^n$ . More precisely, we say that a deterministic protocol  $\Pi_n$  solves the compression game of  $h_n$  with error at most  $\gamma(n) \in [0, 1]$  if

$$\Pr_{x \sim \{0, 1\}^n} [h_n(x) = 1 \iff \text{transcript}_{\Pi_n}(x) \in E_n] \geq 1 - \gamma(n).$$

These definitions are extended to languages in the natural way. Since in this paper all circuit classes are non-uniform, any probabilistic protocol for a language  $L$  with error at most  $\gamma(n)$  can be converted into an average-case protocol with error at most  $\gamma(n)$  (simply by fixing the randomness of Alice in order to minimize the error probability over  $\{0, 1\}^n$ ).

**Interacting with several Bobs.** We discuss here a more general family of multi-party compression games that allow Alice to interact with multiple Bobs during a single round of the game. The different Bobs are not allowed to communicate with each other, only with Alice. The definition of round complexity for such games is slightly different than for standard 2-party compression games. The reason is as follows. For 2-party games, we can assume that the game concludes with a message to Bob, as Bob is all-powerful and can determine the result of the protocol from the final message. In the case of multi-party games, this assumption isn't well motivated, as no individual Bob might have access to all the information about the protocol. It makes more sense to say the game for a Boolean function  $h$  concludes with Alice computing whether  $h(x) = 1$ , where  $x$  is her input. Thus, a 1-round game will consist of Alice sending messages to the various Bobs, the Bobs responding, and finally Alice computing the answer. This naturally extends to a definition of  $r$ -round games.

We will also measure the cost of a protocol somewhat differently. We will again count only the communication from Alice to Bob, but the cost of a protocol will not be the sum of the lengths of all messages sent by Alice. Instead, we will define the cost of a round to be the maximum length of a message sent by Alice to some Bob, and then the cost of the protocol to be the sum of the costs over all rounds. This definition of protocol cost is motivated by the connection of our model with lower bounds on oracle circuits, which we elaborate later. A formal definition is presented below.

Let  $\mathcal{C}$  be a circuit class, and  $k = k(n), r = r(n)$  be arbitrary functions. A  $\mathcal{C}$ -bounded  $(k + 1)$ -party protocol

$$\Pi_n^{[k]} = \langle D^{(1,1)}, \dots, D^{(1,k)}; D^{(2,1)}, \dots, D^{(2,k)}; \dots; D^{(r+1,1)}, \\ g^{(1,1)}, \dots, g^{(r,1)}; g^{(1,2)}, \dots, g^{(r,2)}; \dots; g^{(1,k)}, \dots, g^{(r,k)} \rangle$$

with  $r$  rounds consists of a sequence of  $\mathcal{C}$ -circuits for Alice, and strategies for each Bob $_i$ , given by  $g^{(1,i)} \dots g^{(r,i)}$ . We associate to every  $k$ -party protocol  $\Pi_n^{[k]}$  its signature  $\text{signature}(\Pi_n^{[k]}) = (n, s_1, t_1, \dots, s_r, t_r)$ , where for each  $j \in [r], i \in [k]$ ,  $s_j$  is the maximum length of a message sent by Alice to any Bob $_i$  during the  $j$ -th round, and  $t_j$  is the maximum length of a message sent by any Bob $_i$  to Alice during the  $j$ -th round. For every  $i \in [r], j \in [k]$ ,  $D^{(i,j)}$  maps the sequence of the input  $x$ , all messages sent to Alice before the  $i$ -th round and all of Alice's messages before the  $i$ -th round to Alice's message in the  $j$ -th round to Bob $_j$ .  $D^{(r+1,1)}$  maps the sequence of  $x$  and all messages sent during the protocol to a single bit. For every  $i \in [r], j \in [k]$ ,  $g^{(i,j)}$  maps the sequence of all Alice's messages to Bob $_j$  from the first to the  $i$ -th round to Bob $_j$ 's message to Alice in the  $i$ -th round. We say that  $\Pi_n^{[k]}$  solves the compression game for a function  $h_n$  on  $n$  bits if  $D^{(r+1,1)}$  outputs 1 on  $x$  if and only if  $h_n(x) = 1$ .

Finally, we let  $\text{cost}(\Pi_n^{[k]}) \stackrel{\text{def}}{=} s$ , where  $s = \sum_{i \in [r]} s_i$ . We assume for convenience that the number of parties is always limited by the size of the circuits used by Alice. These definitions extend to languages, probabilistic games, and average-case games in the natural way.

### 3 The communication cost of $\text{AC}^0[p]$ -compression games

We start with a construction of single-round compression games for an arbitrary symmetric function.

► **Lemma 3.1.** *Let  $f: \{0, 1\}^n \rightarrow \{0, 1\}$  be an arbitrary symmetric function. Then, for every  $1 \leq d(n) \leq \log n / \log \log n$ , the function  $f$  admits a single-round  $\text{AC}_d^0(\text{poly}(n))$ -compression game with communication*

$$c_d(n) = O\left((d-1)! \cdot n \cdot \left(\frac{\log \log n}{\log n}\right)^{d-1}\right).$$

In particular, for every fixed integer  $d \geq 1$ , we have  $c_d(n) = O(n/(\log n)^{(d-1)-o(1)})$ .

**Proof.** Let  $f$  be a symmetric function that receives as input an  $n$ -bit string  $x \in \{0, 1\}^{[n]}$ . We sketch the construction of depth- $d$  circuits for the corresponding compression games. Observe that any integer  $n \in \mathbb{N}$  can be represented with at most  $\lceil \log(n+1) \rceil$  bits. For simplicity, we will approximate these values by  $\log n$ . This will be compensated by the use of asymptotic notation in the final bounds.

Observe that for  $d = 1$  the result is obvious, since Alice can simply send  $x$  to Bob. For every  $d \geq 2$ , we design an  $\text{AC}_d^0(\text{poly}(n))$  circuit that, on a given input  $x$ , outputs  $m_d \stackrel{\text{def}}{=} (d-2)! \cdot n \cdot (\log \log n)^{d-2} / (\log n)^{d-1}$  binary strings  $a_d^1, \dots, a_d^{m_d}$  of size  $s_d \stackrel{\text{def}}{=} (d-1) \cdot \log \log n$ , which together encode the number of 1's in  $x$ . More precisely,  $|x|_1 = \sum_{i=1}^{m_d} \text{dec}(a_d^i)$ , where  $\text{dec}(a)$  denotes the integer encoded by the binary string  $a$ . Therefore, it is enough that Alice communicates in a single-round at most  $m_d \cdot s_d$  bits to Bob, which is then able to compute the original value  $f(x)$ . This last step relies on the assumption that  $f$  is a symmetric function.

First, we give a depth-2 circuit with these properties. Partition the  $n$  input bits into  $m_2 = n / \log n$  blocks of size  $t = \log n$ . In other words, let  $[n] = B_1 \dot{\cup} \dots \dot{\cup} B_{m_2}$ , where  $|B_i| = t$ . For each block  $B_i$ , there exists CNFs  $\phi_1^i, \dots, \phi_{\log \log n}^i$  of size  $O(n)$  that compute the string  $a_2^i \in \{0, 1\}^{\log \log n} = \{0, 1\}^{s_2}$  corresponding to the number of 1's in  $x_{B_i} \in \{0, 1\}^{B_i}$  (the projection of  $x$  to  $B_i$ ). A small formula of this form exists because the number of input bits is  $\log n$ . Together with the previous discussion, this completes the proof for  $d = 2$ .

Now fix an arbitrary  $d > 2$ . We will construct the corresponding  $\text{AC}_d^0$  circuit by induction. It will be clear from the description that its final size is a polynomial whose leading exponent



does not depend on  $d$ . Assume that there is a depth  $d - 1$  circuit  $C$  that outputs  $m_{d-1}$  strings  $a_{d-1}^1, \dots, a_{d-1}^{m_{d-1}}$ , as described before, on any given input  $x \in \{0, 1\}^n$ . Assume also that its top gates are AND gates. This is without loss of generality, given the argument we use below.

Recall that  $a_{d-1}^i \in \{0, 1\}^{s_{d-1}}$ . We partition these strings into  $m_d$  sets, each containing  $t \stackrel{\text{def}}{=} m_{d-1}/m_d = \log n / ((d-2) \cdot \log \log n) \geq 1$  strings, given our upper bound on  $d$ . More precisely, we have  $[m_{d-1}] = T_1 \dot{\cup} \dots \dot{\cup} T_{m_d}$ , where  $|T_i| = t$ . For convenience, let  $A_i = \{a_{d-1}^j \mid j \in T_i\}$ . For any  $a_{d-1}^j$ , we have  $\text{dec}(a_{d-1}^j) \leq 2^{s_{d-1}} = (\log n)^{d-2}$ . Consequently,

$$\sum_{j \in A_i} \text{dec}(a_{d-1}^j) \leq |A_i| \cdot (\log n)^{d-2} = t \cdot (\log n)^{d-2} \leq (\log n)^{d-1}.$$

In particular, this sum can be represented with  $s_d = (d-1) \cdot \log \log n$  bits. Observe that the strings in  $A_i$  have, together,  $t \cdot s_{d-1} = \log n$  bits. Therefore, there exists DNFs  $\psi_1^i, \dots, \psi_{s_d}^i$  of size  $O(n)$  that compute the sum of the strings in  $A_i$ , which we represent as a string  $a_d^i \in \{0, 1\}^{s_d}$ . Since this is the case for every  $i \in [m_d]$ , we obtain circuits  $\psi^i \circ C$  computing each string  $a_d^i$ . Finally, notice that the top three layers of  $\psi_j^i \circ C$  can be collapsed into a depth-2 circuit, which gives us an  $\text{AC}_d^0$  circuit for the same function. This completes the proof of Lemma 3.1.  $\blacktriangleleft$

Notice that this upper bound comes from a very restricted class of compression games, as there is no continuing interaction with Bob. A simpler and more efficient construction can be obtained for the  $\text{MOD}_q$  functions, as for them there is no need to keep track of the exact number of 1s in the original input.

As observed by [13], any compression game for  $\text{Majority}_{2n}$  can be used to solve the compression game for  $\text{Parity}_n$ , with some overhead. In general, the same argument provides the following connection, which implies that in order to prove lower bounds for  $\text{Majority}$ , it is sufficient to get lower bounds for  $\text{MOD}_q$ .

**► Proposition 3.2.** *Let  $h: \{0, 1\}^n \rightarrow \{0, 1\}$  be an arbitrary symmetric function,  $\mathcal{C}$  be a circuit class, and  $d \geq 1$ . Assume that the  $\mathcal{C}_d(\text{poly}(n))$ -compression game for  $\text{Majority}_n$  can be solved with cost  $c(n)$  in  $r(n)$  rounds. Then the  $\mathcal{C}_{d+O(1)}(\text{poly}(n))$ -compression game for  $h$  can be solved with cost  $c_h(n) = O(c(2n) \cdot \log n)$  in  $r_h(n) = O(r(2n) \cdot \log n)$  rounds.*

**Proof.** Let  $\Pi_{2n}^{\text{Maj}}$  be a protocol for  $\text{Majority}_{2n}$ . We sketch the construction of a protocol  $\Pi_n^h$  for  $h$ . The idea is to run  $\Pi_{2n}^{\text{Maj}}$  about  $\log n$  times in order to obtain the hamming weight  $|x|_1$  of  $x \in \{0, 1\}^n$ , the input given to Alice in the compression game for  $h$ .

In order to achieve this, Alice runs  $\Pi_{2n}^{\text{Maj}}$  on appropriate inputs of the form  $y = x1^k0^{n-k} \in \{0, 1\}^{2n}$ , where a different  $k$  is used during each stage of  $\Pi_n^h$ . Here a stage is simply a complete execution of  $\Pi_{2n}^{\text{Maj}}$ , and Alice performs a binary search with at most  $O(\log n)$  stages to obtain  $|x|_1$ . Although we have defined protocols with an implicit set  $E$  of accepting transcripts, observe that with an extra round we can ensure that Bob sends the correct output  $\text{Majority}_{2n}(y)$  to Alice.

Finally, it is enough to verify that each string  $y$  can be computed by constant-depth polynomial size circuits. However, since there are no more than  $O(\log n)$  stages, and since Bob sends one bit at each stage, each string  $y$  is a function of at most  $O(\log n)$  bits, and can certainly be computed by depth-two polynomial size circuits.  $\blacktriangleleft$

For our main theorem, we will need the following result, whose proof is discussed in more detail in Section B.



► **Proposition 3.3** ([36, 40], folklore). *Let  $p, q \geq 2$  be distinct primes. There exist fixed constants  $\zeta > 0$  and  $n_0 \in \mathbb{N}$  for which the following holds. For every  $n \geq n_0$  and  $\varepsilon(n) \in [2^{-n}, 1/10q]$ , any polynomial  $P \in \mathbb{F}_p[x_1, \dots, x_n]$  that  $\varepsilon$ -approximates the  $\text{MOD}_q^n$  function with respect to the uniform distribution has degree at least  $\zeta \cdot \sqrt{n \cdot \log(1/\varepsilon)}$ .*

Interestingly, our argument relies on a crucial way on the approximation of Boolean circuits by polynomials with exponentially small error. For convenience of the reader, we include the proof of the next result in Section C.

► **Proposition 3.4** ([36, 40, 29]). *Let  $p$  be a fixed prime. There exists a constant  $\alpha = \alpha(p) \in \mathbb{N}$  such that, for every  $\delta \in (0, 1/2)$  and  $d(n) \geq 1$ , any  $\text{AC}_d^0[p](s(n))$  circuit  $C$  admits a  $\delta$ -error probabilistic polynomial  $\mathbf{Q}(x_1, \dots, x_n) \in \mathbb{F}_p[x_1, \dots, x_n]$  of degree at most  $(\alpha \cdot \log s)^{d-1} \cdot \log(1/\delta)$ . In particular, it follows that for any distribution  $\mathcal{D}$  over  $\{0, 1\}^n$ ,  $C$  is  $\delta$ -approximated with respect to  $\mathcal{D}$  by a polynomial of degree at most  $(\alpha \cdot \log s)^{d-1} \cdot \log(1/\delta)$ .*

The next proposition is a minor extension of a result implicit in [13]. It allows us to transform an interactive compression protocol for a function into a certain Boolean circuit that computes the same function.

► **Proposition 3.5.** *Let  $c: \mathbb{N} \rightarrow \mathbb{N}$  be a function such that  $c(n) \leq n$ ,  $s: \mathbb{N} \rightarrow \mathbb{N}$  be a function with  $s(n) = \Omega(n)$ ,  $\gamma: \mathbb{N} \rightarrow [0, 1/2)$ ,  $L$  be a language, and  $\mathcal{C}$  be a circuit class. If there exists an average-case  $\mathcal{C}_d(\text{poly}(n))$ -compression game for  $L$  with cost  $c(n)$  and error probability  $\gamma(n)$  with respect to the uniform distribution over  $\{0, 1\}^n$ , then there exist circuits  $C_1, \dots, C_T$  from  $\mathcal{C}_{d+O(1)}(\text{poly}(n))$ , where  $T \leq 2^{c(n)}$ , such that*

$$\Pr_{x \sim \{0,1\}^n} [L(x) \neq \bigvee_{i \in [T]} C_i(x)] \leq \gamma(n).$$

Furthermore, these circuits are disjoint:  $C_i^{-1}(1) \cap C_j^{-1}(1) = \emptyset$  for every pair  $i, j \in [T]$  with  $i \neq j$ .

**Proof.** Let  $\Pi_n = \langle C^{(1)}, \dots, C^{(r)}, f^{(1)}, \dots, f^{(r-1)}, E_n \rangle$  be an average-case protocol for  $L_n$  with  $r(n)$  rounds and error probability  $\gamma(n)$ . Observe that  $\Pi_n$  solves the  $\mathcal{C}$ -compression game of some function  $h_n: \{0, 1\}^n \rightarrow \{0, 1\}$ , and that  $h_n$  is  $\gamma(n)$ -close to  $L_n$ . Recall that  $\Pi_n$  has a signature  $\text{signature}(\Pi_n) = (n, s_1, t_1, \dots, t_{r-1}, s_r)$ . For convenience, let  $t \stackrel{\text{def}}{=} \sum_{i \in [r-1]} t_i$ , and  $s \stackrel{\text{def}}{=} c(n) = \sum_{i \in [r]} s_i$ .

Given a string  $w \in \{0, 1\}^{s+t}$ , we write  $w = (w^{(A,1)}, w^{(B,1)}, \dots, w^{(B,r-1)}, w^{(A,r)})$  as a concatenation of strings whose sizes respect the signature of  $\Pi_n$ . In other words,  $|w^{(A,i)}| = s_i$  and  $|w^{(B,j)}| = t_j$ , for all  $i \in [r]$  and  $j \in [r-1]$ . We say that  $w$  is Alice-consistent on an input  $x$  if, for every  $i \in [r]$ ,  $w^{(A,i)} = C^{(i)}(x, w^{(A,1)}, w^{(B,1)}, \dots, w^{(B,i-1)})$ . On the other hand, we say that  $w$  is Bob-consistent if, for every  $j \in [r-1]$ ,  $w^{(B,j)} = f^{(j)}(w^{(A,1)}, \dots, w^{(A,j-1)})$ . Observe that whether a string  $w$  is Bob-consistent or not does not depend on  $x$ . Let  $B_n \subseteq \{0, 1\}^{t+s}$  denote the set of Bob-consistent strings. For convenience, set  $W_n \stackrel{\text{def}}{=} E_n \cap B_n$ .

We claim that  $h(x) = 1$  if and only if there exists a string  $w \in W_n$  that is Alice-consistent on  $x$ . One direction is clear, since if  $h(x) = 1$  then  $\text{transcript}_{\Pi_n}(x) \in E_n$ , and this string is both Bob-consistent and Alice-consistent on  $x$ . On the other hand, assume there exists  $w \in \{0, 1\}^{s+t}$  that is Bob-consistent and Alice-consistent on  $x$ . An easy induction on the number of rounds of the protocol shows that  $w = \text{transcript}_{\Pi_n}(x)$ . Furthermore, if  $w \in W_n$  then  $w \in E_n$ , and it must be the case that  $h(x) = 1$ , since  $\Pi_n$  is a protocol for  $h_n$ . Observe that this argument also shows that if  $h(x) = 1$  then there is a unique  $w \in W_n$  that serves as a certificate for  $x$ .

Notice that there are at most  $2^{c(n)}$  Bob-consistent strings. This is because for every string  $w^A = (w^{(A,1)}, w^{(A,2)}, \dots, w^{(A,r)}) \in \{0,1\}^s$ , there exists a unique completion of  $w^A$  by a string  $w \in \{0,1\}^{s+t}$  that is Bob-consistent. In particular,  $|W_n| \leq 2^{c(n)}$ .

For every fixed  $w \in W_n$ , we claim that there exists a circuit  $C_w(x)$  from  $\mathcal{C}_{d+O(1)}(\text{poly}(n))$  that checks if  $w$  is Alice-consistent on  $x$ . Recall that for every  $i \in [r]$ ,  $C^{(i)}$  is a circuit from  $\mathcal{C}_d(\text{poly}(n))$ . Therefore, we can check in parallel whether

$$w^{(A,i)} = C^{(i)}(w^{(A,1)}, w^{(B,1)}, \dots, w^{(B,i-1)})$$

for all  $i \in [r]$  using just a constant number of additional layers, since we assume throughout that  $\mathcal{C}$  has unbounded fan-in AND and OR gates. which proves the claim. It follows that

$$h(x) = \bigvee_{w \in W_n} C_w(x),$$

for every  $x \in \{0,1\}^n$ . In addition,  $C_{w_1}$  and  $C_{w_2}$  are disjoint whenever  $w_1 \neq w_2$ , since exactly one  $w \in W_n$  is Alice-consistent on  $x$ . Finally, recall that  $h_n$  is  $\gamma(n)$ -close to  $L_n$ , which completes the proof of Proposition 3.5.  $\blacktriangleleft$

Proposition 3.5 implies that in order to prove communication lower bounds for interactive compression games, it is enough to prove circuit lower bounds of a particular form. We obtain the following result.

**► Lemma 3.6.** *Let  $p$  and  $q$  be distinct primes,  $\gamma: \mathbb{N} \rightarrow (0,1)$  be an arbitrary function,  $k \in \mathbb{N}$ , and  $d = d(n) \in \mathbb{N}$ . Assume that*

$$\Pr_{x \sim \{0,1\}^n} [\text{MOD}_q^n(x) \neq \bigvee_{i \in [T(n)]} C_i(x)] \leq \gamma(n),$$

where each  $C_i$  is computed by an  $\text{AC}_d^0[p](n^k)$  circuit, and  $C_i$  and  $C_j$  are disjoint whenever  $i \neq j$ . Then, the following holds.

- (i)  $\log T(n) \geq \sqrt{n}/(\log n)^{d+O(1)}$  if  $\gamma(n) \leq 1/20q$ ;
- (ii)  $\log T(n) \geq n/(\log n)^{2d+O(1)}$  in the case of an exact compression game (i.e.,  $\gamma = 0$ ).

**Proof.** We employ the polynomial approximation method, i.e., we show that if  $\text{MOD}_q^n$  admits a circuit of this form, then it can be approximated by a polynomial  $Q$  whose degree is upper bounded by a function depending on  $T$ . We then invoke Proposition 3.3 in order to obtain a lower bound on  $T$ . More details follow.

First, Proposition 3.4 guarantees that for any  $\delta > 0$ , each circuit  $C_i$  can be  $\delta$ -approximated under the uniform distribution by a polynomial  $Q_i \in \mathbb{F}_p[x_1, \dots, x_n]$  of degree at most  $(\ell \cdot \log n)^d \cdot \log(1/\delta)$ , where  $\ell$  is a fixed positive constant. We let  $\delta \stackrel{\text{def}}{=} \varepsilon/T$ , where  $\varepsilon = \varepsilon(n)$  will be set conveniently later in the proof. Now let

$$Q(x) \stackrel{\text{def}}{=} \sum_{i \in [T]} Q_i(x).$$

We claim that  $Q \in \mathbb{F}_p[x_1, \dots, x_n]$  is a polynomial that  $(\varepsilon + \gamma)$ -approximates  $\text{MOD}_q^n$  under the uniform distribution. Clearly,

$$\begin{aligned} \Pr_{x \sim \{0,1\}^n} [\text{MOD}_q^n(x) \neq Q(x)] &\leq \Pr \left[ \text{MOD}_q^n(x) \neq \bigvee_{i \in [T(n)]} C_i(x) \right] + \Pr \left[ \bigvee_{i \in [T(n)]} C_i(x) \neq Q(x) \right] \\ &\leq \gamma + \left( 1 - \Pr \left[ \bigvee_{i \in [T(n)]} C_i(x) = Q(x) \right] \right). \end{aligned}$$

For each  $i \in [T]$ , let  $S_i \stackrel{\text{def}}{=} \{x \in \{0, 1\}^n \mid Q_i(x) \neq C_i(x)\}$  be the set of bad inputs for  $Q_i$ , and set  $S \stackrel{\text{def}}{=} \bigcup_{i \in [T]} S_i$ . In order to complete the proof of our claim, we argue next that for every  $y \notin S$ ,  $Q(y) = \bigvee_{i \in [T(n)]} C_i(y)$ .

First, if  $\bigvee_{i \in [T(n)]} C_i(y) = 0$ , then  $Q_i(y) = 0$  for every  $i \in [T]$ , and we get  $Q(y) = 0$ . On the other hand, if  $\bigvee_{i \in [T(n)]} C_i(y) = 1$ , using the disjointness assumption for the family of circuits, it follows that there is exactly one circuit with  $C_i(y) = 1$ . Since  $y \notin S$ , we get that  $Q_i(y) = 1$ , while  $Q_j(y) = 0$  for every  $j \neq i$ . Consequently, we have  $Q(y) = 1$ . (Observe that the extra assumption over the family of circuits is crucial for this case, since the original circuits produce Boolean values, while  $Q$  is an  $\mathbb{F}_p$ -polynomial.) Overall, it follows that  $\Pr[\bigvee_{i \in [T(n)]} C_i(x) = Q(x)] \geq (2^n - |S|) \cdot 2^{-n} \geq 1 - T \cdot \delta = 1 - \varepsilon$ , which establishes our initial claim.

Therefore, for every  $\varepsilon(n) > 0$ , there exists a polynomial  $Q \in \mathbb{F}_p[x_1, \dots, x_n]$  that  $(\varepsilon + \gamma)$ -approximates the  $\text{MOD}_q^n$  function over the uniform distribution, where

$$\deg(Q) \leq ((\ell \cdot \log n)^d \cdot \log(1/\delta)) \leq (\ell \cdot \log n)^d \cdot (\log T + \log(1/\varepsilon)). \quad (1)$$

On the other hand, we obtain from Proposition 3.3 that for every  $\varepsilon(n) \in [2^{-n}, 1/10q]$ , and every large enough  $n$ ,

$$\zeta \cdot \sqrt{n \cdot \log(1/(\varepsilon + \gamma))} \leq \deg(Q). \quad (2)$$

Our result follows by combining Equations 1 and 2. Observe that we are free to set  $\varepsilon(n)$  in order to maximize our lower bound on  $T$ , depending on the value of  $\gamma$ . If  $0 < \gamma \leq 1/20q$ , the first case of Lemma 3.6 follows if we let  $\varepsilon = 1/20q$ . On the other hand, when  $\gamma = 0$ , we get that

$$\log T(n) \geq \frac{\zeta \cdot \sqrt{n \cdot \log(1/\varepsilon)} - \log(1/\varepsilon) \cdot (\ell \cdot \log n)^d}{(\ell \cdot \log n)^d},$$

and the second case of Lemma 3.6 now follows by setting  $\varepsilon = \exp(-\Theta(n/\log^{2d} n))$ . ◀

We are now ready to prove an essentially optimal communication lower bound for  $\text{AC}_d^0[p]$ -compression games for Majority.

▶ **Theorem 3.7.** *Let  $p$  be a prime number. There exists a constant  $c \in \mathbb{N}$  such that, for any  $d \in \mathbb{N}$ , and every  $n \in \mathbb{N}$  sufficiently large, the following holds.*

- (i) *Any  $\text{AC}_d^0[p]$ -compression game for Majority $_n$  (with any number of rounds) has communication cost at least  $n/(\log n)^{2d+c}$ .*
- (ii) *There exists a single-round  $\text{AC}_d^0$ -compression game for Majority $_n$  with communication cost at most  $n/(\log n)^{d-c}$ .*

**Proof.** The lower bound follows immediately from Proposition 3.2, Proposition 3.5, and Lemma 3.6 (ii). The upper bound is given by Lemma 3.1. ◀

For randomized compression games, we are able to generalize the lower bound for single-round protocols obtained by Chattopadhyay and Santhanam [13] to protocols with any number of rounds.

▶ **Theorem 3.8.** *Let  $p$  and  $q$  be distinct primes. There exists a constant  $c \in \mathbb{N}$  such that, for any  $d \in \mathbb{N}$ , and  $n \in \mathbb{N}$  sufficiently large, every randomized  $\text{AC}_d^0[p]$ -compression game for  $\text{MOD}_q^n$  with any number of rounds and error at most  $1/3$  has communication cost at least  $\sqrt{n}/(\log n)^{d+c}$ .*

**Proof.** If there exists a randomized compression protocol with these properties, we can boost its success probability to  $1 - 1/20q$  on every input by repeating it a constant number of times, and applying a majority vote. Observe that the communication increases by a constant factor only, and that the majority vote can be computed efficiently, as it is over a constant number of bits. Since any randomized protocol with this success probability provides an average-case protocol that is correct on at least a  $(1 - 1/20q)$ -fraction of the inputs under the uniform distribution, the result follows from Proposition 3.5 and Lemma 3.6 (i). ◀

We stress that the results in Theorems 3.7 and 3.8 hold both for Majority and  $\text{MOD}_q$ , but we restricted each statement to a particular function for simplicity. In order to see this, first notice that the proof of Theorem 3.7 includes the argument for  $\text{MOD}_q$ . On the other hand, in order to extend Theorem 3.8 to Majority, we can employ a reduction through Proposition 3.2. A subtle point is that for probabilistic protocols one has to make sure that the final error probability after the reduction is bounded. However, this can be achieved during the proof by boosting the correctness probability of the initial protocol for Majority via repetition.

The proof of Theorem 3.7 can be generalized to an essentially optimal bound for  $\text{AC}_d^0[p](s(n))$ -compression games computing  $\text{MOD}_q^n$ . The argument implies that this function has communication cost  $n/(\log s)^{\Theta(d)}$ . Observe that the original circuit size lower bounds obtained by Razborov [36] and Smolensky [40] follows from the analysis of communication protocols for Majority and  $\text{MOD}_q$  with constant communication cost. Interestingly, the polynomial method interpolates between essentially optimal communication lower bounds and circuit size lower bounds when applied with exponentially small error and constant error, respectively.

## 4 Multiparty Interactive Compression

### 4.1 The communication cost of $k$ -party $\text{AC}^0[p]$ -compression games

We will prove in this section that  $\text{Majority}_n$  requires  $\tilde{\Omega}(n^{1/2r})$  communication in the  $(k + 1)$ -party  $r$ -round  $\text{AC}^0[p]$ -compression game, for any  $k = \text{poly}(n)$ . Put another way, although Alice is allowed to send roughly  $n^{1/2r}$  bits to each individual Bob, even if  $n^{100}$  such parties are present, she will not be able to combine their answers in order to compute  $\text{Majority}_n$ .

We start with the following upper bound, which can be seen as the corresponding analogue of Lemma 3.1.

► **Lemma 4.1.** *Let  $f: \{0, 1\}^n \rightarrow \{0, 1\}$  be an arbitrary symmetric function, and  $p$  be any prime. For any  $r \in \mathbb{N}$ ,  $f$  admits an  $(\lceil n^{1/r} \rceil + 1)$ -party  $r$ -round  $\text{AC}^0$ -compression game with cost  $O(rn^{1/r} \log(n))$ .*

**Proof.** We set up some notation first. Given  $n$  and  $r$ , let  $T_{n,r}$  be the complete  $\lceil n^{1/r} \rceil$ -ary tree of depth  $r$ . We assume the leaves of  $T_{n,r}$  to be ordered from left to right. Given an input  $x$  of length  $n$ , label the leaves of  $T_{n,r}$  with bits of  $x$  in the natural way: the leftmost leaf is labelled with the first bit of  $x$ , the second to leftmost with the second bit, etc. Note that some leaves may remain unlabelled in this process.

Let  $V_d$  be the set of nodes at depth  $d$  in this tree, where  $0 \leq d \leq r$ . The protocol will proceed with Alice iteratively labelling nodes in the tree with numbers in  $[n]$ , each node being labelled with the sum of all the leaves in the subtree rooted at the node. Any unlabelled leaf is assumed to have label 0. After round  $i$ , where  $0 \leq i \leq r$ , all nodes at depth  $r - i$  or greater will be labelled. Once the root is labelled, Alice can compute  $f(x)$  by herself, as  $f(x)$

is purely a function of the label at the root (which is the weight of the input  $x$ ), and any function of  $O(\log n)$  bits can be computed in  $\text{AC}_2^0$ .

We assume inductively that after round  $i$ , all nodes at depth  $r - i$  or greater have been labelled. The base case  $i = 0$  clearly holds, as Alice can label the leaves herself. Assume that the inductive hypothesis holds after round  $i$ , where  $0 \leq i < r$ . We show it holds after round  $i + 1$ . In round  $i + 1$ , Alice arbitrarily associates a unique Bob with each node  $v \in V_{r-i-1}$ . This can be done as long as the number of parties is greater than  $\lceil n^{1/r} \rceil$ , as assumed. We denote the Bob associated with  $v$  by  $\text{Bob}(v)$ . For each  $v$ , Alice sends to  $\text{Bob}(v)$  the sequence of labels of the children of  $v$ . Note that by the inductive assumption, the children of  $v$  have already been labelled. For each  $v$ ,  $\text{Bob}(v)$  responds with the sum of all the integer labels sent by Alice to  $\text{Bob}(v)$  in the  $(i + 1)$ -th round.

This is clearly a correct protocol. In any one round, Alice sends at most  $\lceil n^{1/r} \rceil \cdot \lceil \log(n+1) \rceil$  bits to any Bob, as the number of children of any node in the tree is at most  $\lceil n^{1/r} \rceil$ , and each labelled node has a label in  $[n]$ . Thus, the cost of the protocol is  $O(rn^{1/r} \log n)$ , as claimed.  $\blacktriangleleft$

Our lower bound is also based on algebraic arguments, but it employs a slightly different approach to that in the previous section. In particular, it does not rely on Proposition 3.5. We will need the following result.

► **Proposition 4.2** ([36]). *Let  $p$  be a fixed prime, and  $P(x_1, \dots, x_n) \in \mathbb{F}_p[x_1, \dots, x_n]$  be a degree- $\ell$  polynomial. Then,*

$$\Pr_{x \sim \{0,1\}^n} [\text{Majority}_n(x) = P(x)] \leq 1/2 + O(\ell/\sqrt{n}).$$

The next lemma allows us to construct low-degree probabilistic polynomials from multi-party compression games.

► **Lemma 4.3.** *Let  $\Phi_n^{[k]}$  be a randomized  $(k + 1)$ -party  $r$ -round  $\text{AC}_d^0[p](\text{poly}(n))$ -compression protocol with signature  $(n, s_1, t_1, \dots, s_r, t_r)$  computing a Boolean function  $h: \{0, 1\}^n \rightarrow \{0, 1\}$  with error  $\gamma$ , where  $s_i \leq n$  for each  $i \in [r]$ , and  $r \in \mathbb{N}$ . Then, for every  $\delta > 0$ ,  $h$  admits a  $(\gamma + \delta)$ -error probabilistic polynomial over  $\mathbb{F}_p$  with degree  $O((\sum_{i \in [r]} s_i)^r \cdot ((\log n)^{d+r} \cdot (\log 1/\delta))^{r+1})$ .*

**Proof.** We start with a proof of the lemma for  $r = 1$  and deterministic protocols that are always correct, then observe that the same proof can be generalized to randomized  $r$ -round protocols.

Suppose  $\Phi_n^{[k]}$  is a  $(k + 1)$ -party 1-round  $\text{AC}_d^0[p](\text{poly}(n))$ -compression protocol with signature  $(n, s_1, t_1)$  for a Boolean function  $h$  on inputs  $x$  of  $n$  bits. For each  $i \in [k]$ , let  $a_1^i \dots a_{n_i}^i$  be the message bits sent by Alice to  $\text{Bob}_i$  in the first round, and let  $b_1^i \dots b_{m_i}^i$  be  $\text{Bob}_i$ 's response. Let  $a$  be the bit output by Alice at the conclusion of the protocol. By the definition of signature, we have that for each  $i \in [k]$ ,  $n_i \leq s_1$  and  $m_i \leq t_1$ . We also have that  $a = 1$  if and only if  $h(x) = 1$ .

Each of the message bits sent by Alice in the first round is a function of  $x$ , and since Alice is  $\text{AC}_d^0[p](\text{poly}(n))$ -bounded, we can use Proposition 3.4 to obtain  $\varepsilon$ -error probabilistic polynomials  $P_j^i \in \mathbb{F}_p[x_1, \dots, x_n]$ , where  $i \in [k]$ ,  $j \in [n_i]$ , for each of these message bits. The degree of each polynomial is at most  $d_1 = O((\log n)^{d-1} \cdot \log 1/\varepsilon)$ , where  $\varepsilon > 0$  is a parameter to be determined later. Since each message bit of each  $\text{Bob}_i$  is a function of the message bits sent by Alice to  $\text{Bob}_i$ , we can express each bit  $b_j^i$  of  $\text{Bob}_i$  as an *exact* polynomial  $Q_j^i$  in the message bits of Alice. Notice that each such polynomial has degree at most  $s_1$ . Now,

again by Proposition 3.4, there is an  $\varepsilon$ -error probabilistic polynomial  $P$  of degree at most  $d_2 = O((\log n)^{d-1} \cdot \log 1/\varepsilon)$  for  $a$  as a function of  $x$ , the message bits sent by Alice in the first round, and the message bits sent by each Bob in the first round.

If we set  $\varepsilon = \delta/(s_1 \cdot k + 1)$ , by using the union bound, we have that

$$P' \stackrel{\text{def}}{=} P(x, P_1^1(x), \dots, P_{n_k}^k(x), Q_1^1(P_1^1(x), \dots, P_{n_1}^1(x)), \dots, Q_{m_k}^k(P_1^k(x), \dots, P_{n_k}^k(x)))$$

is a  $\delta$ -error probabilistic polynomial for  $h$  as a function of  $x$ . The degree of  $P'$  is at most  $d_1 \cdot s_1 \cdot d_2 = O(s_1 \cdot ((\log n)^d \cdot \log 1/\delta)^2)$ , where we have used that  $\log 1/\varepsilon = O(\log n \cdot \log 1/\delta)$  due to the upper bound on  $s_1$  and  $k \leq \text{poly}(n)$ . This completes the proof for (deterministic) single-round protocols.

The proof for deterministic protocols with  $r \geq 2$  rounds is by induction on the number of rounds. Let  $\Phi_n^{[k]}$  be a  $(k+1)$ -party  $r$ -round  $\text{AC}_d^0[p](\text{poly}(n))$ -compression protocol with signature  $(n, s_1, t_1, \dots, s_r, t_r)$  for a Boolean function  $h$ . Observe that during the last round of the protocol, each  $\text{Bob}_\ell$  receives a message containing at most  $s \stackrel{\text{def}}{=} \sum_{i \in [r]} s_i$  bits (recall that  $\text{Bob}_\ell$  has access to the messages he received from Alice in previous rounds, and to no other message). We can view each bit  $a_j^\ell$  of each such message as a Boolean function computed by a  $(k+1)$ -party  $(r-1)$ -round protocol, where  $\ell \in [k]$ , and  $j \leq s$ . It follows from the induction hypothesis that there is a probabilistic polynomial  $P_j^\ell \in \mathbb{F}_p[z_1, \dots, z_{s'}]$  for an appropriate  $s' \leq s$  of degree at most

$$d_1 \leq O(s^{r-1} \cdot ((\log n)^{d+(r-1)} \cdot (\log 1/\varepsilon))^r)$$

that  $\varepsilon$ -approximates  $a_j^\ell$ , where  $\varepsilon > 0$  will be set conveniently later in the proof.<sup>2</sup> Further, during the last round of the protocol, each bit  $b_j^\ell$  sent by  $\text{Bob}_\ell$  can be computed exactly by a (deterministic) polynomial  $Q_j^\ell$  of degree at most  $s$ . Finally, the last bit output by Alice during the execution of  $\Phi_n^{[k]}$  is computed by an  $\text{AC}_d^0[p]$  circuit over polynomially many input bits. According to Proposition 3.4, it can be  $\varepsilon$ -approximated by a probabilistic polynomial  $P \in \mathbb{F}_p[y_1, \dots, y_{\text{poly}(n)}]$  of degree  $d_2 \leq O((\log n)^{d-1} \cdot \log 1/\varepsilon)$ .

We now compose these polynomials appropriately, similarly to the base case, in order to obtain a probabilistic polynomial  $P' \in \mathbb{F}_p[x_1, \dots, x_n]$  that approximates the original Boolean function  $h$  compressed by  $\Phi_n^{[k]}$ . If we set  $\varepsilon \stackrel{\text{def}}{=} \delta/(sk + 1) = \delta/\text{poly}(n)$ , we get via an union bound that  $P'$  is a probabilistic polynomial that  $\delta$ -approximates  $h$ . Finally, the degree of  $P'$  is upper bounded by

$$\begin{aligned} d_1 \cdot s \cdot d_2 &\leq O(s^{r-1} \cdot ((\log n)^{d+(r-1)} \cdot (\log 1/\varepsilon))^r \cdot s \cdot (\log n)^{d-1} \cdot \log 1/\varepsilon) \\ &\leq O(s^r \cdot ((\log n)^{d+r} \cdot (\log 1/\delta))^r \cdot (\log n)^d \cdot \log 1/\delta) \\ &\leq O((\sum_{i \in [r]} s_i)^r \cdot ((\log n)^{d+r} \cdot (\log 1/\delta))^{r+1}), \end{aligned}$$

which completes the induction step.

It remains to handle the case of randomized protocols. Observe that for every fixed setting of the randomness of Alice, we obtain a multiparty compression protocol computing some Boolean function  $h_r$ . We can apply the procedure described above to get a probabilistic polynomial  $P_r \in \mathbb{F}_p[x_1, \dots, x_n]$  that agrees with  $h_r$  on every input  $x \in \{0, 1\}^n$  except with probability  $\delta$ . Since over the choice of  $r$  we know that  $h(x) = h_r(x)$  except with probability  $\gamma$ , we can obtain from the family of distributions  $P_r$  a single distribution over polynomials of

<sup>2</sup> Our abuse of the asymptotic notation in this inductive proof is harmless, as we are proving the result for a fixed number of rounds only.



the same degree that agrees with  $h$  on every input  $x$  except with probability  $\gamma + \delta$ , which completes the proof.  $\blacktriangleleft$

We now have all ingredients to prove the main result of this section.

► **Theorem 4.4.** *Let  $p \in \mathbb{N}$  be a fixed prime. For every  $k, r, d \in \mathbb{N}$ , the following holds.*

- (i) *There exists a deterministic  $n^{1/r}$ -party  $r$ -round  $\text{AC}^0[p]$ -compression game for  $\text{Majority}_n$  with cost  $O(n^{1/r} \cdot \log n)$ .*
- (ii) *Every randomized  $n^k$ -party  $r$ -round  $\text{AC}_d^0[p]$ -compression game for  $\text{Majority}_n$  has cost  $\Omega(n^{1/2r} / (\log n)^{2(d+r)})$ .*

**Proof.** The upper bound follows from Lemma 4.1. For the lower bound, assume  $\Pi_n^{[k]}$  has signature  $(n, s_1, t_1, \dots, s_r, t_r)$  and satisfies the assumption of the theorem. Since  $\Pi_n^{[k]}$  is a randomized protocol, we can reduce its error probability to  $1/20$  by running it in parallel and computing a majority vote during the last round. Observe that the depth of the circuits used by Alice increases by at most 1 if this computation is performed by an appropriate DNF or CNF. Setting  $\delta = 1/20$  in Lemma 4.3 and fixing the randomness, we can obtain an average-case (deterministic) polynomial for  $\text{Majority}_n$  of the stated degree and error  $1/10$  with respect to the uniform distribution. Now applying Proposition 4.2 and using  $1/\delta = O(1)$ , we get that

$$(s_1 + s_2 + \dots + s_r)^r \cdot (\log n)^{(d+r)(r+1)} \geq \Omega(\sqrt{n}),$$

which completes the proof of the lower bound, since  $\text{cost}(\Pi_n^{[k]}) = \sum_{i \in [r]} s_i$  and  $r \geq 1$ .  $\blacktriangleleft$

As opposed to the statement of Theorem 3.7, we have not tried to optimize the logarithmic factors here, since there is still a polynomial gap in the bounds as a function of  $r$ .<sup>3</sup>

► **Corollary 4.5.** *For any  $r, \ell, d \in \mathbb{N}$ , the randomized  $n^\ell$ -party  $r$ -round  $\text{AC}_d^0[p]$ -compression cost of  $\text{Majority}_n$  is  $n^{\Theta(1/r)}$ .*

In addition, observe that Theorem 4.4 implies a round separation result for *multiparty*  $\text{AC}^0[p]$ -compression games. In particular, we get the following consequence for single-round  $\text{AC}^0[p]$  protocols versus protocols with more rounds.

► **Corollary 4.6.** *For every  $\varepsilon > 0$  and  $\ell \in \mathbb{N}$ , there exists  $r \in \mathbb{N}$  with  $r = O(1/\varepsilon)$  for which the following holds, whenever  $n$  is sufficiently large. There exists an explicit function  $f_n: \{0, 1\}^n \rightarrow \{0, 1\}$  such that:  $f_n$  admits no randomized  $n^\ell$ -party single-round  $\text{AC}^0[p]$ -compression games with cost  $n^{1/2-\varepsilon}$ , but it admits deterministic  $n^\varepsilon$ -party  $r$ -round  $\text{AC}^0[p]$ -compression games of cost  $n^\varepsilon$ .*

## 4.2 Randomized versus deterministic games

Note that for two-party games we were able to obtain almost linear lower bounds for *deterministic* protocols (Theorem 3.7), while for probabilistic and average-case protocols we encountered a barrier at  $c(n) \approx \sqrt{n}$  (Theorems 3.8 and 4.4). We are not aware of explicit lower bounds of the form  $n^{1/2+\varepsilon}$  for a fixed  $\varepsilon > 0$  for randomized two-party  $\text{AC}^0[p]$  games. It is natural to wonder if we can improve Theorem 4.4 in the case of *deterministic*  $k$ -party games.

<sup>3</sup> For instance, in the proof of Lemma 4.1, it is possible to break the information passed to each Bob into multiple blocks as done in the proof of Lemma 3.1, and save an extra  $(\log n)^{\Theta(d)}$  factor during each round by allowing Alice to make partial progress towards the computation of  $\text{Majority}$ .

We prove next that this is unlikely without the introduction of new ideas to handle probabilistic protocols. More precisely, we observe that  $k$ -party protocols can be derandomized without increasing communication cost. The proof relies on the definition of cost for such protocols as the length of the longest message sent by Alice to any particular Bob, and on the fact that we are dealing with non-uniform protocols/circuits. The argument is based on parallel repetition and composition of  $k$ -party protocols with an approximate majority function. We provide the details next.

We say that a Boolean function  $h_n: \{0, 1\}^n \rightarrow \{0, 1\}$  is an  $(\ell_1, \ell_2)$ -approximate majority if  $h_n(x) = 0$  on every  $x$  with  $|x|_1 \leq \ell_1$ , and  $h_n(x) = 1$  on every  $x$  with  $|x|_1 \geq \ell_2$ .

► **Proposition 4.7** ([3]). *There exists a family  $h = \{h_n\}_{n \in \mathbb{N}}$  of Boolean functions in  $\text{AC}_3^0(\text{poly}(n))$  for which every  $h_n$  is an  $(0.49n, 0.51n)$ -approximate majority.*

► **Theorem 4.8.** *Let  $\mathcal{C}$  be a circuit class,  $d \geq 1$ , and  $f = \{f_n\}_{n \in \mathbb{N}}$  be a family of Boolean functions, where  $f_n: \{0, 1\}^n \rightarrow \{0, 1\}$ . Suppose  $f$  admits a  $k$ -party probabilistic  $\mathcal{C}_d(\text{poly}(n))$ -compression game with cost  $c(n)$  and error  $\gamma(n) \leq 1/3$ , where  $k = O(\text{poly}(n))$ . Then  $f$  admits a  $k'$ -party deterministic  $\mathcal{C}_{d+O(1)}(\text{poly}(n))$ -compression game with the same cost  $c(n)$  and  $k' = O(\text{poly}(n))$ .*

**Proof.** By assumption,  $f$  has a  $k$ -party probabilistic  $\mathcal{C}_d(\text{poly}(n))$ -compression protocol  $\Pi$  with cost  $c(n)$  and error  $\gamma(n) \leq 1/3$ , where  $k = O(\text{poly}(n))$ . We define a new probabilistic protocol for  $f$  with the same cost but with  $k' \stackrel{\text{def}}{=} \ell n \cdot k$  parties and with error  $\gamma'(n) < 2^{-n}$ , where  $\ell > 0$  is a constant which we determine later. We then use Adleman's trick to fix the random bits used by Alice, thus making the protocol deterministic.

The new probabilistic protocol  $\Pi'$  for  $f$  simply simulates  $\ell n$  copies of the protocol  $\Pi$  in parallel. Namely, we interpret the Bobs to be partitioned into  $\ell n$  sets, each of size  $k$ , and Alice independently executes the protocol in parallel for each set of Bobs. Note that by our definition of cost, the cost for each round of  $\Pi'$  is the same as the cost for each round of  $\Pi$ . In the final step of the protocol,  $\Pi'$  applies the Approximate Majority function  $h_{\ell n}$  to the answers of  $\Pi$  for the  $\ell n$  parallel executions. Using Proposition 4.7, Alice can be implemented to work in  $\mathcal{C}_{d+O(1)}(\text{poly}(n))$ . It follows by a standard application of Proposition 1.1 that if we set  $\ell$  to be a large enough constant, the error probability of the new protocol  $\Pi'$  is strictly less than  $2^{-n}$ .

Now, there must exist some setting of the random bits of Alice that yields the correct answer for every  $x \in \{0, 1\}^n$ , simply by using the union bound. By fixing the random bits of Alice accordingly, we derive a *deterministic* protocol with cost  $c(n)$ , which completes the proof. ◀

## 5 The connection with circuits augmented with oracle gates

In this section we observe that lower bounds on interactive compressibility are closely connected to lower bounds against oracle circuits with arbitrary oracles. We first show such a connection for 2-party compression games, and then for multiparty compression games.

In order to formalize these connections, we need to define classes of oracle circuits corresponding to classes of Boolean circuits. Such a definition is especially non-obvious for bounded-depth circuit classes – should we consider oracle gates when counting the depth or not? We use a very generous notion of oracle circuits. We say that an oracle circuit  $C$  belongs to the oracle analogue of a Boolean circuit class  $\mathcal{C}$  if every maximal subcircuit of  $C$  without oracle gates belongs to  $\mathcal{C}$ . Put another way, every subcircuit induced by a connected subgraph of the acyclic graph encoding  $C$  that does not contain an oracle gate is



a circuit from  $\mathcal{C}$ . The generosity of this notion only makes the lower bounds we derive from the connections below stronger.

For the sake of convenience, we abuse notation and occasionally use  $\mathcal{C}$  to refer both to a Boolean circuit class and its oracle analogue.

► **Proposition 5.1.** *Let  $\mathcal{C}$  be a circuit class. Let  $C$  be an oracle circuit over  $n$  variables from  $\mathcal{C}(\text{poly}(n))$  with oracle gates  $f_i: \{0, 1\}^{s_i} \rightarrow \{0, 1\}^{t_i}$ , where  $i \in [r]$ , for some  $r = r(n)$ . In addition, let  $s = s_1 + \dots + s_r$  be the total fan-in of these oracle gates, and  $h: \{0, 1\}^n \rightarrow \{0, 1\}$  be the Boolean function computed by  $C$ . Then  $h$  admits a  $\mathcal{C}(\text{poly}(n))$ -compression game with communication cost  $c(n) \leq s + 1$  consisting of at most  $r + 1$  rounds.*

**Proof.** We describe a protocol for the compression game for  $h$  in which Alice sends at most  $s + 1$  bits to Bob, and where each of Alice's messages is computable by a small circuit from  $\mathcal{C}$ .

First Alice topologically sorts the circuit  $C$  with respect to oracle gates, namely she constructs a graph  $G$  whose nodes are the oracle gates of the circuit, and there is an edge from a node  $u$  to a node  $v$  if and only if there is a path from the oracle gate represented by  $u$  to the oracle gate represented by  $v$  in the digraph  $C$ . The graph  $G$  is a DAG, and hence its vertices can be topologically sorted. Let  $g_1, g_2 \dots g_r$  be the topological ordering of the oracle gates. Alice proceeds inductively as follows. In round  $i$ , where  $i \in [r]$ , she computes all inputs to the gate  $g_i$  using her input  $x$  and previous messages sent by Bob. She then sends the values of these input bits to Bob, who in turn computes the value of the gate  $g_i$  applied to these bits, and sends her the answer. Note that  $g_1$  has no predecessors which are oracle gates, and therefore Alice can compute all the inputs to  $g_1$  herself using circuits from  $\mathcal{C}$  (which are sub-circuits of  $C$ ) applied to the input  $x$ . Gate  $g_i$  only has gates  $g_1 \dots g_{i-1}$  as predecessors, and by the definition of the protocol, Alice has already received the values of these gates from Bob in previous rounds, hence she can calculate values of inputs to  $g_i$  from  $x$  and previous messages using circuits from  $\mathcal{C}$ . In round  $r + 1$ , Alice computes the value of the circuit  $C$  on  $x$  and sends it to Bob, thus completing the protocol.

The total number of bits sent by Alice to Bob is the total fan-in of the oracle gates plus one, i.e.,  $s + 1$ , and there are  $r + 1$  rounds in the protocol. ◀

Note that Proposition 5.1 only gives useful information when the total fan-in of oracle gates is sub-linear. We'd like to also show lower bounds on oracle gates where the total fan-in is not bounded in this way. This is where multiparty compression games, and the modified notion of protocol cost for such games, come in useful.

We need some more terminology for oracle circuits. An oracle circuit  $C$  has  $r$  layers if the oracle gates can be partitioned into  $r$  sets such that no two gates within any set are connected by a path in  $C$ . Equivalently, there are at most  $r$  oracle gates on any path from an input of  $C$  to the output.

► **Proposition 5.2.** *Let  $D$  be an oracle circuit over  $n$  variables from  $\mathcal{C}(\text{poly}(n))$  augmented with  $r$  layers of oracle gates, where for each  $i \in [r]$ ,  $s_i$  is the maximum fan-in of a gate in the  $i$ -th layer, and where there are at most  $k$  gates in each layer. Let  $s = \sum_{i \in [r]} s_i$ . In addition, let  $h: \{0, 1\}^n \rightarrow \{0, 1\}$  be the Boolean function computed by  $D$ . Then  $h$  admits a  $(k + 1)$ -party  $\mathcal{C}(\text{poly}(n))$ -compression game with  $r$  rounds and communication cost  $c(n) \leq s$ .*

**Proof.** Alice orders the layers of oracle gates topologically, so that there are no paths from gates in layer  $i$  to gates in layer  $j$  for  $i > j$ . The protocol proceeds with Alice inductively computing all input bits to oracle gates in the  $i$ -th layer, where  $i \in [r]$ , and then delegating the computations of gates in the  $i$ -th layer to the Bobs, a different Bob for each oracle gate. Since there are at most  $k$  gates in each such layer, she can successfully assign a different Bob

to each oracle gate in any specific layer. Alice can compute all inputs to an oracle gate in the first layer by herself, as all of these can be computed by circuits in  $\mathcal{C}(\text{poly}(n))$ . In the  $i$ -th round, where  $i \in [r]$ , Alice chooses a different Bob for each oracle gate in layer  $i$ , and sends to the corresponding Bob the values of the inputs to the corresponding gate. She can compute these values using circuits in  $\mathcal{C}$ , as the output bits of all oracle gates in layer  $i - 1$  or below are already known to her by the definition of the protocol. The Bob corresponding to a gate responds with the output values of that gate. After the  $r$ -th round, Alice computes the output value of the circuit  $C$ , and outputs it.

Notice that Alice sends at most  $s_i$  bits to any individual Bob in round  $i$  by our assumption on the fan-in of oracle gates in  $C$ . Thus the cost of the protocol is  $s$ . It is clear that the protocol operates in  $r$  rounds. ◀

Observe that Propositions 5.1 and 5.2, together with Theorems 3.7 and 4.4, imply strong limitations on the progress that  $\text{AC}^0[p]$  circuits can make towards the goal of computing the Majority function. In particular, a circuit of this form extended with arbitrary oracle gates can only compute  $\text{Majority}_n$  if it delegates essentially all the work to these extra gates. We can formalize this claim as follows.

► **Corollary 5.3.** *Let  $p \geq 2$  be prime, and  $d \in \mathbb{N}$ . There exists a constant  $c \in \mathbb{N}$  such that, for every sufficiently large  $n$ , the following holds. If  $\text{Majority}_n$  is computed by  $\text{AC}_d^0[p]$  circuits of polynomial size with arbitrary oracle gates, then the total fan-in of the oracle gates is at least  $n/(\log n)^{2d+c}$ .*

**Proof.** This result follows immediately from Proposition 5.1 and Theorem 3.7. The fan-in lower bound is independent of the number of oracle gates, as Theorem 3.7 holds for protocols with any number of rounds. ◀

This result has an interesting consequence on the structure of  $\text{AC}^0[p]$  circuits computing Majority. More precisely, Corollary 5.3 implies that in any layered circuit computing  $\text{Majority}_n$ , at least  $\lfloor n/(\log n)^{O(k)} \rfloor$  gates must be present at the  $k$ -th layer of the circuit (in order to see this, transform the circuit into an equivalent circuit with a single oracle gate at the top after the first  $k$  layers). On the other hand, the construction in Lemma 3.1 shows that this bound is not far from optimal. A similar consequence holds for polynomial size circuits computing the  $\text{MOD}_q$  function.

Using Proposition 5.2 and Theorem 4.4, we derive lower bounds on the maximum fan-in of oracle gates in oracle circuits with a bounded number of such layers computing Majority. The number of oracle gates is now allowed to be polynomially large.

► **Corollary 5.4.** *Let  $p \geq 2$  be prime, and  $r, d \in \mathbb{N}$ . If  $\text{Majority}_n$  is computed by an  $\text{AC}_d^0[p]$  circuit of polynomial size with arbitrary oracle gates that contains at most  $r$  layers of such gates, then there is some oracle gate with fan-in at least  $n^{1/2r}/\text{polylog}(n)$ .*

Proposition 5.2 suggests an approach to the NP vs.  $\text{NC}^1/\text{poly}$  problem. The key observation is that for any  $r$ , every Boolean function in  $\text{NC}^1/\text{poly}$  has oracle circuits of polynomial size with  $r$  layers, where the maximum fan-in of any oracle gate is  $n^{O(1/r)}$ .

► **Proposition 5.5.** *Let  $f = \{f_n\}_{n \in \mathbb{N}}$  be a family of Boolean functions in  $\text{NC}^1/\text{poly}$ , and  $r \in \mathbb{N}$ . Then  $f$  has  $\text{AC}^0$  oracle circuits of polynomial size with  $r$  layers, where the maximum fan-in of any oracle gate is  $n^{O(1/r)}$ .*

**Proof.** Let  $\{C_n\}_{n \in \mathbb{N}}$  be a sequence of circuits for  $f$ , where each  $C_n$  has size at most  $n^k$  and depth at most  $c \log n$ , for fixed constants  $k$  and  $c$ . We define oracle circuits  $D_n$  as follows.

Divide  $C_n$  into  $r$  equally spaced layers of gates, with the distance between any two layers being at most  $(c/r) \log n$ . Replace each node at a layer boundary by an oracle gate whose inputs are its predecessors on the previous layer boundary. Note that any oracle gate has at most  $n^{c/r}$  inputs, since the circuit has bounded fan-in. There are clearly a polynomially bounded number of oracle gates. Also, the circuit is an  $AC^0$  circuit, since it consists purely of inputs and oracle gates. ◀

Applying Proposition 5.2 yields the following corollary.

▶ **Corollary 5.6.** *Let  $r$  be any positive integer. Every function in  $NC^1/\text{poly}$  admits  $\text{poly}(n)$ -party  $AC^0(\text{poly}(n))$ -compression games with  $r$  rounds and cost  $n^{O(1/r)}$ .*

Thus a stronger lower bound than in Corollary 5.4 for an explicit function in NP would imply a separation of NP and  $NC^1/\text{poly}$ . We conjecture that Clique is such a function.

## 6 Interactive Compression versus Computation

The results of this paper and in [13] show that two important techniques in circuit complexity, namely, random restrictions and approximation by low-degree polynomials, can be used to prove strong incompressibility lower bounds. It is natural to wonder if other important lower bounds in complexity theory can be extended in a similar way. A related problem is whether compression can be easier than exact computation. Our next result sheds more light into these questions.

Let  $IP_n: \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$  be the Inner Product function. In other words, for  $x, y \in \{0, 1\}^n$ ,  $IP_n(x, y) \stackrel{\text{def}}{=} \sum_{i \in [n]} x_i \cdot y_i \pmod{2}$ . It is known that  $IP_n \notin \text{THR} \circ \text{MAJ}$ , i.e., this function cannot be computed by polynomial size circuits consisting of a bottom layer of linear threshold functions with polynomial weights, connected to a top gate computed by an arbitrary linear threshold function ([20, 21]).<sup>4</sup>

We observe below that  $IP_n$  admits a  $(\text{MAJ} \circ \text{MAJ})$ -compression game with communication cost  $O(\log n)$ . In other words, there is a natural Boolean function that cannot be computed by certain circuits, but whose computation becomes feasible if Alice is allowed to interact with a more powerful party.

▶ **Proposition 6.1.** *Let  $IP = \{IP_n\}_{n \in \mathbb{N}}$  be the family of Inner Product functions. There exists a  $(\text{MAJ} \circ \text{MAJ})$ -compression game for IP with communication cost  $c(n) = O(\log n)$ .*

**Proof.** The protocol consists of  $O(\log n)$  rounds, where in each round Alice sends a single bit, and Bob replies with a string  $v \in \{0, 1\}^n$ . After the last round, Bob knows the sum  $\sum_{i \in [n]} x_i \cdot y_i$ , and therefore the transcript reveals the value  $IP_n(x, y)$ . More details follow.

Alice's circuits are of the form  $C(x, y, v)$ . In the first layer of the circuit,  $C$  computes  $z_i \stackrel{\text{def}}{=} x_i \wedge y_i$ , for every  $i \in [n]$ . In the second layer,  $C$  outputs  $\text{sign}(\sum_{i \in [n]} z_i - v_i)$ . Put another way, Alice uses the same circuit in every round, and we assume that the first bit sent by Alice during the first round is discarded. Bob does all the work, and simulates a binary search by sending to Alice an appropriate string  $v$  during each round. For instance, Bob sends  $v = 0^{n/2}1^{n/2}$  during the first round, and the next bit computed by Alice reveals if  $\sum_{i \in [n]} x_i \cdot y_i$  is at least  $n/2$ . After each round, Bob sends a string corresponding to the next step of the binary search, and so on. Clearly, after  $O(\log n)$  rounds, Bob knows the value  $\sum_{i \in [n]} x_i \cdot y_i$ . Finally, observe that Alice communicates  $O(\log n)$  bits, and that her circuits are of the form  $\text{MAJ} \circ \text{MAJ}$ . ◀

<sup>4</sup> Recall that a function  $f: \{0, 1\}^n \rightarrow \{0, 1\}$  is a linear threshold function if there exist weights  $w_1, \dots, w_n \in \mathbb{Z}$  and a threshold  $\theta \in \mathbb{Z}$  such that  $f(x) = \text{sign}(\sum_{i \in [n]} w_i \cdot x_i - \theta)$ .

## 7

 An improved round separation theorem for  $AC^0$ 

Recall that Chattopadhyay and Santhanam [13] proved that there are Boolean functions on  $n$  variables that admit  $AC^0$ -bounded protocols with  $r$  rounds and cost  $O(n^{1/r})$ , but for which any correct  $AC^0$ -bounded  $(r - 1)$ -round protocol has cost  $\Omega(n^{2/r - o(1)})$ . We use a different construction and refine their techniques, obtaining the following result.

► **Theorem 7.1.** *Let  $r \geq 2$  and  $\varepsilon > 0$  be fixed parameters. There is an explicit family of functions  $f = \{f_n\}_{n \in \mathbb{N}}$  with the following properties:*

- (i) *There exists an  $AC_2^0(n)$ -bounded protocol  $\Pi_n$  for  $f_n$  with  $r$  rounds and cost  $c(n) \leq n^\varepsilon$ , for every  $n \geq n_f$ , where  $n_f$  is a fixed constant that depends on  $f$ .*
- (ii) *Any  $AC^0(\text{poly}(n))$ -bounded protocol  $\Pi$  for  $f$  with  $r - 1$  rounds has cost  $c(n) \geq n^{1-\varepsilon}$ , for every  $n \geq n_\Pi$ , where  $n_\Pi$  is a fixed constant that depends on  $\Pi$ .*

We will need some additional definitions and notation in order to establish this result. For any  $n \in \mathbb{N}$ , let  $g_n: \{0, 1\}^n \rightarrow \{0, 1\}$  be the parity function on  $n$  variables, and  $g = \{g_n\}_{n \in \mathbb{N}}$ . Let  $m, \ell$ , and  $r$  be positive integers. Set  $n = n(m, \ell, r) \stackrel{\text{def}}{=} m + \ell \cdot r \cdot m$ . We define a function  $f_{m, \ell, r}: \{0, 1\}^n \rightarrow \{0, 1\}$  that will be used to prove round separation results for  $AC^0$ -compression games. For convenience, let  $k \stackrel{\text{def}}{=} \log \ell$  and  $v \stackrel{\text{def}}{=} m / \log \ell$ . The definition of  $f_{m, \ell, r}$  depends on  $g$  and a given function  $h: \{0, 1\}^k \rightarrow [\ell]$ , which we assume to be some fixed one-to-one function.

Given any string  $z \in \{0, 1\}^n$ , we write  $z = (x, y^{(\cdot, 1)}, \dots, y^{(\cdot, r)})$ , where  $x \in \{0, 1\}^m$ , and  $y^{(\cdot, j)} = (y^{(1, j)}, \dots, y^{(\ell, j)})$ , where  $j \in [r]$ , and  $y^{(i, j)} \in \{0, 1\}^m$ , for every  $i \in [\ell]$ . In addition, for any string  $w \in \{0, 1\}^m$ , we write  $w = (w^{(1)}, \dots, w^{(k)})$ , where each  $w^{(u)} \in \{0, 1\}^v$ , for  $u \in [k]$ . For convenience, instead of writing  $y^{(i, j)(u)}$ , we may also use  $y^{(i, j, u)}$ .

The function  $f_{m, \ell, r}$  is defined by induction on  $r$ . It is simply a pointer jumping function, where  $h$  is applied to certain bits computed from the current string (initially  $x$ ) using  $k = \log \ell$  independent applications of  $g_v$ . After jumping from the initial  $x$  to a new string  $x'$ , which will be one of the  $y$ 's in  $y^{(\cdot, 1)}$ , we recurse. After  $r$  steps, some string  $y$  from  $y^{(\cdot, r)}$  will be reached. The output of  $f_{m, \ell, r}$  is then set to be  $g_m(y)$ .

Formally, when  $r = 1$ , for any  $z \in \{0, 1\}^n$ ,

$$f_{m, \ell, 1}(z) \stackrel{\text{def}}{=} g_m(y^{(i, 1)}), \text{ where } i = h(g_v(x^{(1)}), \dots, g_v(x^{(k)})).$$

Now let  $r \geq 2$  be arbitrary. Then, for any  $z \in \{0, 1\}^n$ ,

$$f_{m, \ell, r}(z) \stackrel{\text{def}}{=} f_{m, \ell, r-1}(z'),$$

where  $z' = (x', y^{(\cdot, 2)}, \dots, y^{(\cdot, r)})$ ,  $x' = y^{(i, 1)}$ , and  $i = h(g_v(x^{(1)}), \dots, g_v(x^{(k)}))$ . This completes the definition of  $f_{m, \ell, r}$ .

► **Lemma 7.2 (Upper Bound).** *For any  $m, \ell, r \geq 1$ , the function  $f_{m, \ell, r}$  admits an  $AC_2^0(m \cdot \ell)$ -compression game with  $r + 1$  rounds and communication cost  $c(n) = (r + 1) \cdot m$ .*

**Proof.** During each round  $j$ , Alice sends her current string  $x' \in \{0, 1\}^m$  to Bob, which replies with  $\ell$  strings  $v^{(i)} \in \{0, 1\}^m$  satisfying the following property:  $v^{(i)} = 1^m$  if the next round of the game is played on  $y^{(i, j+1)}$ , and  $v^{(i)} = 0^m$  otherwise. Observe that the next message that Alice has to send is simply the  $m$ -bit string given by

$$\bigvee_{i \in [\ell]} (v^{(i)} \wedge y^{(i, j+1)}).$$

The cost and round complexity of this protocol is clear. ◀

We now proceed with the proof that in any  $\text{AC}^0$ -bounded protocol for  $f_{m,\ell,r}$  with  $r$  rounds, Alice has to communicate roughly  $\ell \cdot m$  bits, for an appropriate choice of  $\ell$  that we would like to make as large as possible. The argument is based on random restrictions, which allow us to simplify the  $\text{AC}^0$  circuits used by Alice considerably, while still maintaining the resulting function sufficiently hard for compression games. At a high level, we apply a round elimination technique, combined with a strong lower bound for  $f_{m,\ell,1}$ . More details follow.

From now on we will also view  $f_{m,\ell,r}$  as a function  $f_{m,\ell,r}: \{0,1\}^{[n]} \rightarrow \{0,1\}$ , where each input  $z$  for  $f_{m,\ell,r}$  can also be interpreted as a function  $z: [n] \rightarrow \{0,1\}$ . This will give us more flexibility when manipulating restrictions. A *restriction*  $\rho \in \{0,1,*\}^{[n]}$  is simply a function  $\rho: [n] \rightarrow \{0,1,*\}$ . Given a restriction  $\rho$  and a function  $f: \{0,1\}^{[n]} \rightarrow \{0,1\}$ , we let  $f^\rho: \{0,1\}^{\rho^{-1}(*)} \rightarrow \{0,1\}$  be the following function. For every  $z^- \in \{0,1\}^{\rho^{-1}(*)}$ ,

$$f^\rho(z^-) \stackrel{\text{def}}{=} f(z),$$

where  $z \in \{0,1\}^{[n]}$  is the function with  $z|_{\rho^{-1}(\{*\})} = z^-$  and  $z|_{\rho^{-1}(\{0,1\})} = \rho|_{\rho^{-1}(\{0,1\})}$ .

Let  $N \stackrel{\text{def}}{=} [n]$ . Recall that we write  $z \in \{0,1\}^n$  as  $z = (x, y^{(1,1)}, \dots, y^{(\ell,r)})$ . Similarly, we let  $S^{(i,j,u)} \subseteq N$  index the variables corresponding to  $y^{(i,j,u)}$ , for  $i \in [\ell]$ ,  $j \in [r]$  and  $u \in [k]$ . We define  $S^{(i,j)} \stackrel{\text{def}}{=} \bigcup_u S^{(i,j,u)}$ . Further, we use  $M \subseteq N$  to index the variables corresponding to  $x$ , and  $M^{(1)}, \dots, M^{(k)}$  for the corresponding variables  $x^{(1)}, \dots, x^{(k)}$ . Let  $\Gamma_N$  be the set of all restrictions with domain  $N$ , i.e.,  $\Gamma_N \stackrel{\text{def}}{=} \{0,1,*\}^N$ . Given  $\rho_1, \rho_2 \in \Gamma_N$ , we say that  $\rho_2$  extends  $\rho_1$  if  $\rho_2^{-1}(*) \subseteq \rho_1^{-1}(*)$  and  $\rho_2|_{\rho_1^{-1}(\{0,1\})} = \rho_1|_{\rho_1^{-1}(\{0,1\})}$ .

Our round separation theorem will be derived from lower bounds on a class of functions  $\phi_{s,d,\ell}: \mathbb{N} \times \mathbb{N} \times \mathbb{R}^+ \rightarrow \mathbb{N}$ , defined as follows:

$$\phi_{s,d,\ell}(m, r, \delta) \stackrel{\text{def}}{=} \min_{\sigma \in \Gamma_{N,\delta}} \min_{\Pi \in \text{Prot}_{s,d,r}^\sigma} \text{cost}(\Pi),$$

where:<sup>5</sup>

- (i)  $\Gamma_{N,\delta} \subseteq \Gamma_N$  is the set of all restrictions  $\sigma$  for which the following holds: there exists sets  $D_j \subseteq [\ell]$  with  $j \in [r]$  such that  $|D_j| \leq \delta \cdot \ell$ , and  $\sigma^{-1}(\{0,1\}) = \bigcup_{j \in [r]} \left( \bigcup_{i \in D_j} S^{(i,j)} \right)$ ,
- (ii)  $\text{Prot}_{s,d,r}^\sigma$  is the set of all  $\text{AC}_d^0(s)$ -bounded  $r$ -round protocols  $\Pi$  solving the compression game of  $f_{m,\ell,r}^\sigma$ .

The parameters  $m$ ,  $r$ , and  $\delta$  will vary during our inductive proof, while  $s$ ,  $d$ , and  $\ell$  remain fixed (observe that this is reflected in our notation for  $\phi$ ). The proof of Theorem 7.1 relies on the following lemmas, whose proof we present later in this section.

► **Lemma 7.3** (Lower Bound: Base case). *Let  $s = n^{c_1}$ ,  $d \in \mathbb{N}$ ,  $\ell = m^{c_2}$ ,  $\delta \in (0, 1/10)$ , and  $r = 1$ , where  $c_1$  and  $c_2$  are fixed positive integers. Then, for every fixed  $\beta \in (0, 1/10)$  and  $m$  sufficiently large,*

$$\phi_{s,d,\ell}(m, 1, \delta) \geq \ell \cdot m^{1-\beta}.$$

► **Lemma 7.4** (Lower Bound: Induction step). *Let  $s = n^{c_1}$ ,  $d \in \mathbb{N}$ ,  $\ell = m^{c_2}$ ,  $\delta \in (0, 1/10)$ , and  $r \geq 2$ , where  $c_1$  and  $c_2$  are fixed positive integers. Then, for every fixed  $\beta \in (0, 1/10)$  and  $m$  sufficiently large,*

$$\phi_{s,d,\ell}(m, r, \delta) \geq \min \left\{ \ell \cdot m^{1-\beta}, \phi_{s,d,\ell}(m^{1-\beta}, r-1, \delta + \beta) \right\}.$$

<sup>5</sup> For the sake of this proof, we consider circuits of size at most  $s$  (exactly), instead of  $O(s)$ .

These lemmas imply the following result.

► **Proposition 7.5.** *For every fixed  $r \geq 1$ ,  $c \in \mathbb{N}$ , and  $\zeta > 0$ , for  $m$  sufficiently large, we have*

$$\phi_{\text{poly}(n), O(1), m^c}(m, r, 1/(100r)) \geq \ell \cdot m^{1-\zeta}.$$

**Proof.** The result follows easily from Lemmas 7.3 and 7.4 using that  $r$  is constant and that we can take  $\beta$  and  $\delta$  sufficiently small. ◀

Finally, it is not hard to derive the main lower bound of this section from these results.

**Proof of Theorem 7.1.** Given any  $r \geq 2$  and  $\varepsilon > 0$ , it is enough to consider an appropriate family of functions  $f_{m, \ell, r-1}$ , where  $c = c(\varepsilon)$  is sufficiently large, and set  $\ell = m^c$ . The result then follows from Lemma 7.2 and Proposition 7.5. ◀

We proceed now with the proof of the lemmas. We will need the notion of a random restriction. Let  $p \in [0, 1]$  be a real number. We let  $\Gamma_N^p$  denote the distribution over restrictions  $\rho \in \Gamma_N$  generated by independently fixing each  $\rho(i)$  (where  $i \in N$ ) as follows:

$$\Pr[\rho(i) = *] = p, \quad \Pr[\rho(i) = 1] = (1-p)/2, \quad \Pr[\rho(i) = 0] = (1-p)/2.$$

Given a Boolean function  $f_n: \{0, 1\}^n \rightarrow \{0, 1\}$  over  $n$  variables, we let  $\text{DT}_{\text{depth}}(f)$  be the smallest decision tree depth among all decision trees computing  $f_n$ . The next statement is independent of the number of inputs of  $f$ .

► **Lemma 7.6** (Switching Lemma [27]). *Let  $f$  be a Boolean function that can be written as a conjunction or disjunction of any number of depth- $t$  decision trees. Then, for every  $p \in [0, 1]$  and  $r \in \mathbb{N}$ ,*

$$\Pr_{\rho \sim \Gamma^p} [\text{DT}_{\text{depth}}(f^\rho) > r] \leq (5pt)^r.$$

The next result is a standard consequence of Lemma 7.6 (cf. Gopalan and Servedio [24]).

► **Proposition 7.7.** *Let  $f$  be a Boolean function computed by an  $\text{AC}^0$  circuit of size  $M$  and depth  $d$ . For every  $t \in \mathbb{N}$ , if  $p \leq 1/(10t)^d$  then*

$$\Pr_{\rho \sim \Gamma^p} [\text{DT}_{\text{depth}}(f^\rho) > t] \leq M \cdot 2^{-t}.$$

Given a function  $C: \{0, 1\}^{[n]} \rightarrow \{0, 1\}$ , we let  $\text{live}(C) \subseteq [n]$  denote the set of input variables of  $C$  with influence greater than zero. It will be more convenient for us to rely on the following straightforward consequence of Lemma 7.6 and Proposition 7.7.

► **Lemma 7.8.** *Let  $C_1, \dots, C_{s_1}: \{0, 1\}^{n_1} \rightarrow \{0, 1\}$  be functions computed by depth- $d$   $\text{AC}^0$  circuits of size at most  $n_1^{c_1}$ , where  $d, c_1 \in \mathbb{N}$  and  $s_1 = m^{1-\gamma} \cdot \ell$ , and these parameters satisfy  $m, \ell \in \mathbb{N}$ ,  $\gamma \in (0, 1/5)$ ,  $\ell = m^{c_2}$ , where  $c_2 \in \mathbb{N}$ , and  $n_1 = \Theta(m \cdot \ell)$ . Then, for  $p = m^{-\gamma/2}$ , there exists a constant  $c_3$  such that, as  $m \rightarrow \infty$ ,*

$$\Pr_{\rho \sim \Gamma_{[n_1]}^p} \left[ \left| \bigcup_{i \in [s_1]} \text{live}(C_i^\rho) \right| \leq c_3 \cdot (m^{1-\gamma} \cdot \ell) \right] \rightarrow 1.$$

**Proof.** Let  $p = p_1 \cdot p_2$ , where  $p_1 = p_2 = m^{-\gamma/4}$ . Observe that sampling a restriction  $\rho \sim \Gamma_{[n_1]}^p$  is equivalent to first sampling some  $\rho_1 \sim \Gamma_{[n_1]}^{p_1}$ , followed by a restriction  $\rho_2 \sim \Gamma_W^{p_2}$ , where  $W \stackrel{\text{def}}{=} [n_1] \setminus \rho_1^{-1}(\{0, 1\})$ , and finally setting  $\rho = \rho_2 \circ \rho_1$ , where the composition operation



is defined in the natural way. Let  $c = c_1 + 10$ , and  $t = c \cdot \log n_1$ . Furthermore, we let  $r = \lceil 8(1 + c_2)/\gamma \rceil$ , and  $c_3 = 2^r$ . Then,

$$\begin{aligned} \Pr_{\rho \sim \Gamma_{[n_1]}^p} \left[ \left| \bigcup_{i \in [s_1]} \text{live}(C_i^\rho) \right| > c_3 \cdot (m^{1-\gamma} \cdot \ell) \right] &\leq \Pr_{\rho \stackrel{\text{def}}{=} \rho_2 \circ \rho_1} \left[ \exists i \in [s_1] \text{ s.t. } |\text{live}(C_i^\rho)| > 2^r \right] \\ &\leq \Pr_{\rho_1, \rho_2} \left[ \exists i \in [s_1] \text{ s.t. } \text{DT}_{\text{depth}}(C_i^\rho) > r \right] \end{aligned}$$

In order to conclude the proof, it is enough to show that for every  $j \in [s_1]$  and sufficiently large  $m$ ,  $\Pr_{\rho_1, \rho_2}[\text{DT}_{\text{depth}}(C_j^\rho) > r] \leq (1/n_1)^2$ . However, by our choice of parameters (and with room to spare), this follows from an application of Proposition 7.7 with  $\rho_1$  and  $t$ , followed by an application of Lemma 7.6 with  $\rho_2$  and  $r$  (notice that these statements are true with respect to any input size). ◀

We are now ready to prove Lemmas 7.3 and 7.4.

**Proof of Lemma 7.3.** Let  $\sigma: [n] \rightarrow \{0, 1, *\}$  be a restriction in  $\Gamma_{N, \delta}$ , where  $n = m + \ell \cdot m$  and  $N = [n]$ , as usual. Let  $N_1 \stackrel{\text{def}}{=} N \setminus \sigma^{-1}(\{0, 1\})$ , and set  $n_1 \stackrel{\text{def}}{=} |N_1|$ . Observe that  $n_1 \geq (1 - \delta) \cdot \ell \cdot m = \Theta(m \cdot \ell)$ . In addition, let  $\Pi = (C^{(1)}, g^{(1)}, E)$  be a single-round protocol for  $f_{m, \ell, 1}^\sigma$ , where  $C^{(1)} = (C_1, \dots, C_{s_1})$ , and these are  $\text{AC}^0$  circuits of depth  $d$  and size  $s = n^{c_1} \leq n_1^{2c_1}$  (for large enough  $m$ ) that compute the message in  $\{0, 1\}^{s_1}$  that Alice sends to Bob. By definition, for each  $i \in [s_1]$ ,  $C_i: \{0, 1\}^{n_1} \rightarrow \{0, 1\}$ . We prove that if  $s_1 < \ell \cdot m^{1-\beta}$ , then there exists an input  $z \in \{0, 1\}^{n_1}$  for which  $\Pi(z) \neq f_{m, \ell, 1}^\sigma(z)$ .

Let  $D_1 \subseteq [\ell]$  be the set identifying the variables  $y$  fixed by  $\sigma$  (according to our definition of  $\Gamma_{N, \delta}$ ). For any  $z \in \{0, 1\}^{N_1}$ , we write  $z = (x, y^{(i_1, 1)}, \dots, y^{(i_k, 1)})$ , where  $[\ell] \setminus D_1 = \{i_1, \dots, i_k\}$ ,  $k \geq (1 - \delta) \cdot \ell$ , and  $x \in \{0, 1\}^m$ . Recall that we use sets  $S^{(i_1, 1)}, \dots, S^{(i_k, 1)}$  and  $M$  to address the elements of  $[N_1]$  corresponding to these input positions.

Now consider a random restriction  $\rho \sim \Gamma_{N_1}^p$ , where  $p = m^{-\beta/2}$ . Applying Lemma 7.8 with  $\gamma = \beta$  and Proposition 1.1, it follows that, for every large enough  $m$ , with high probability:

- (i)  $C^{(1), \rho}$  depends on at most  $O(m^{1-\beta} \cdot \ell)$  variables.
- (ii) For every  $j \in [\log \ell]$ , it is the case that  $\rho^{-1}(*) \cap M^{(j)} \neq \emptyset$ .
- (iii)  $|\rho^{-1}(*) \cap (S^{(i_1, 1)} \cup \dots \cup S^{(i_k, 1)})| \geq \frac{1}{2} \cdot \frac{(1-\delta) \cdot m \cdot \ell}{m^{\beta/2}} = \Omega(m^{1-\beta/2} \cdot \ell)$ . In particular, from (i) we get that there exists  $i \in [\ell] \setminus D_1$  for which  $S^{(i, 1)} \cap (\rho^{-1}(*) \setminus \text{live}(C^{(1), \rho})) \neq \emptyset$ .

Overall, it follows that there exists a restriction  $\bar{\rho} \in \Gamma_N$  with  $\bar{\rho} = \rho \circ \sigma$ , for an appropriate choice of  $\rho \in \Gamma_{N_1}$ , such that  $\bar{\rho}$  fixes the message sent by Alice, but does not fix the value of  $f_{m, \ell, 1}^{\bar{\rho}}$ . In particular, there exists a  $z \in \{0, 1\}^{n_1}$  that agrees with  $\bar{\rho}$  for which  $\Pi(z) \neq f_{m, \ell, 1}^\sigma(z)$ , which completes the proof. ◀

The proof of Lemma 7.4 is not much harder than the argument used in the base case, but it has a few technicalities that need to be handled.

**Proof of Lemma 7.4.** Let  $\sigma \in \Gamma_{N, \delta}$  and  $\Pi \in \text{Prot}_{s, d, r}^\sigma$  be a pair realizing  $\phi_{s, d, \ell}(m, r, \delta)$ . In other words,  $\Pi$  solves the compression game of  $f_{m, \ell, r}^\sigma$ , and  $\text{cost}(\Pi) = \phi_{s, d, \ell}(m, r, \delta)$ . Assume that  $\Pi = (C^{(1)}, \dots, C^{(r)}, g^{(1)}, \dots, g^{(r-1)}, E)$ , and  $\text{signature}(\Pi) = (n_1, s_1, t_1, \dots, t_{r-1}, s_r)$ , where  $n = m + m \cdot \ell \cdot r$ ,  $N = [n]$ ,  $N_1 = N \setminus \sigma^{-1}(\{0, 1\})$ , and  $n_1 = |N_1|$ . For convenience, let  $C^{(1)} = (C_1, \dots, C_{s_1})$ , where each  $C_i$  is a depth- $d$   $\text{AC}^0$  circuit of size at most  $n^{c_1} \leq n_1^{2c_1}$  (for large  $m$ ), since  $n_1 \geq (1 - \delta) \cdot n$ .

Notice that if  $\text{cost}(\Pi) \geq \ell \cdot m^{1-\beta}$  then the statement of Lemma 7.4 is true. Otherwise, from  $\text{cost}(\Pi) < \ell \cdot m^{1-\beta}$  we get that  $s_1 < \ell \cdot m^{1-\beta}$ , which allows us to proceed as in the proof

of Lemma 7.3. Let  $p = m^{-\beta/2}$ , and set  $\gamma = \beta$ . It follows from Lemma 7.8 that, with high probability,

$$|\text{live}(C^{(1),\rho})| = O(m^{1-\beta} \cdot \ell). \quad (3)$$

Let  $D_j$  for  $j \in [r]$  be the sets identifying the variables  $y$  fixed by  $\sigma$ . By assumption,  $|D_j| \leq \delta \cdot \ell$  for every  $j \in [r]$ . From now on, whenever we consider a set  $S^{(i,j)}$ , we implicitly assume that  $j \in [r]$  and  $i \in [\ell] \setminus D_j$ . This time we will also be concerned about how the action of  $\rho$  affects the more specific sets  $S^{(i,j,u)}$ , where  $u \in [\log \ell]$ . Observe that, with high probability (Proposition 1.1), for every  $(i, j, u)$ , we have:

$$|S^{(i,j,u)} \cap \rho^{-1}(*)| \geq \frac{1}{2} \cdot \frac{m}{\log \ell} \cdot p = \frac{1}{2} \cdot \frac{m^{1-\beta/2}}{c_2 \log m} \geq m^{1-(3/4)\beta}, \quad (4)$$

for any sufficiently large  $m$ . We say that a set  $S^{(i,j)}$  is *bad* with respect to  $C^{(1),\rho}$  if  $|S^{(i,j)} \cap \text{live}(C^{(1),\rho})| \geq \frac{1}{2} \cdot m^{1-(3/4)\beta}$ . Otherwise, the set is said to be *good*. It follows from Equation 3 that

$$\text{Number of bad sets } S^{(i,j)} \leq \frac{O(m^{1-\beta} \cdot \ell)}{(1/2) \cdot m^{1-(3/4)\beta}} = \frac{2\ell}{m^{\beta/4}} = o(\ell), \quad (5)$$

as  $m \rightarrow \infty$ . In particular, since  $r = O(1)$  and  $\beta$  is a fixed constant, with high probability, for every  $j \in [r]$  there are at most  $\beta \cdot \ell$  sets  $S^{(i,j)}$  that are bad with respect to  $C^{(1),\rho}$ . Finally, with high probability over  $\rho$ , we also get that, for every  $j \in [\log \ell]$ ,

$$|M^{(j)} \cap \rho^{-1}(*)| > 0.$$

It follows using the probabilistic method that there exists a fixed restriction  $\rho_1 \in \Gamma_{N_1}$  satisfying all these properties. Let  $\rho_2 = \rho_1 \circ \sigma$  be the restriction obtained by combining  $\rho_1$  and  $\sigma$  in the obvious way. Observe that  $\rho_2: N \rightarrow \{0, 1, *\}$ . Fix arbitrarily all  $*$ -variables in  $\rho_2$  corresponding to bad sets  $S^{(i,j)}$ . On every good set  $S^{(i,j)}$ , fix all  $*$ -variables intersecting  $\text{live}(C^{(1),\rho_1})$ , and also fix additional variables in each set  $S^{(i,j,u)}$  so that the new restriction  $\rho_3$  satisfies  $|\rho_3^{-1}(*) \cap S^{(i,j,u)}| = m^{1-\beta}$ , for every appropriate triple  $(i, j, u)$ . This is possible for any large enough  $m$ , since these sets are good. Further, we assume that the number of variables corresponding to each  $S^{(i,j,u)}$  that are set to 1 is *even*, in order not to invert the parity inside each block, which will be important later in the proof. Let  $f_{m,\ell,r}^{\rho_3}: \{0, 1\}^{\rho_3^{-1}(*)} \rightarrow \{0, 1\}$  be the resulting function.

Given an input  $\tilde{z} \in \{0, 1\}^{\rho_3^{-1}(*)}$ , write  $\tilde{z} = (\tilde{x}, \{\tilde{y}^{(i,j)}\})$ , and let  $z = (x, \{y^{(i,j)}\}) \in \{0, 1\}^n$  be the completion of  $\tilde{z}$  that *agrees* with  $\rho_3$ , where this notion is defined in the natural way. Observe that  $h(x)$  still depends on  $\tilde{x}$ . Now we set all remaining  $*$ -variables in  $M$  in a way that, for the new restriction  $\bar{\sigma}: [N] \rightarrow \{0, 1, *\}$ , we have  $h(\bar{\sigma}(M))$  pointing to a pair  $(i, 1)$  corresponding to a good set  $S^{(i,1)}$ . This is possible due to the properties of  $\rho_1$ . Observe that  $C^{(1),\bar{\sigma}}$  computes a constant function (i.e., Alice's message  $a^{(1)}$  has been fixed). Let  $b^{(1)} \in \{0, 1\}^{\ell_1}$  be the answer provided by Bob, which is also fixed.

Now let  $\bar{\Pi} = (\bar{C}^{(1)}, \dots, \bar{C}^{(r-1)}, \bar{g}^{(1)}, \dots, \bar{g}^{(r-2)}, E)$  be a new protocol obtained by setting each  $\bar{C}^{(i)}$  to be  $C^{(i+1)}$  with its input corresponding to the first message sent by Bob fixed to  $b^{(1)}$ , and  $\bar{g}^{(i)} = g^{(i+1)}$ , for every appropriate  $i$ . If we also rename the input variables in  $f_{m,\ell,r}^{\bar{\sigma}}$  and in the functions and circuits from  $\bar{\Pi}$ , truncating irrelevant variables appropriately (recall the definition of the original function as a pointer jumping function), we obtain a restriction  $\sigma': \{0, 1\}^{N'} \rightarrow \{0, 1\}$ , where  $n' = |N'| = m' + m' \cdot \ell \cdot r'$ ,  $m' = m^{1-\beta}$ ,  $r' = r - 1$ ,  $\sigma' \in \Gamma_{N',\delta'}$ ,  $\delta' = \delta + \beta$ , and the resulting protocol  $\Pi' \in \text{Prot}_{s,d,r'}^{\sigma'}$ . Crucially,



$\Pi'$  is a protocol solving the compression game of  $f_{m',\ell,r'}^{\sigma'}$  in  $r'$  rounds, which implies that  $\text{cost}(\Pi) \geq \text{cost}(\Pi') \geq \phi_{s,d,\ell}(m', r', \delta') = \phi_{s,d,\ell}(m^{1-\beta}, r-1, \delta + \beta)$ , completing the proof of Lemma 7.4.  $\blacktriangleleft$

## 8 Open Problems and Further Research Directions

Our results and techniques raise a number of interesting questions, which we discuss more carefully below.

**The power of interaction in two-party  $\text{AC}^0[p]$ -compression games.** Observe that the approach to obtain communication lower bounds for  $\text{AC}^0[p]$  games employed in the proof of Theorem 1.1 is insensitive to the number of rounds of the protocol. On the other hand, our round separation result (Theorem 1.6) holds with respect to  $\text{AC}^0$  circuits only. Consequently, a natural question is whether a strong round separation theorem is true for  $\text{AC}^0[p]$  games. We conjecture that this is the case, and that a hard function can be obtained via a similar construction that uses  $\text{MOD}_q$  instead of parity.

**Randomized  $\text{AC}^0[p]$ -compression games.** While we have obtained essentially optimal lower bounds for deterministic two-party  $\text{AC}^0[p]$ -compression games, the situation is less clear with respect to randomized protocols. Modulo logarithmic factors, there is a quadratic gap between our upper and lower bounds for  $\text{MOD}_q$  and Majority (Theorem 1.3). On the other hand, it is known that the communication cost of these games is  $n/\log^{\Theta(d)} n$  for randomized  $\text{AC}_d^0$ -compression games (Chattopadhyay and Santhanam [13]). We are unable to obtain better lower bounds here because our approach does not seem to tolerate the initial error probability from the protocol, as it relies on the low error regime of the polynomial approximation method.

**Extending circuit lower bounds to incompressibility results.** The results presented in this paper and in [13] show that recent extensions of the random restriction method and the polynomial approximation method can provide optimal incompressibility results. However, our construction from Section 6 implies that not every technique can be extended in this sense. Which other techniques and results from circuit complexity can be strengthened to compressibility lower bounds?

**Understanding the structure of Boolean circuits.** Our results shed more light into the computation of Boolean functions such as  $\text{MOD}_q$  using  $\text{AC}^0[p]$  circuits, as we are able to obtain information about each layer of the circuit. Similar developments appear for instance in Tarui [42], Rudich and Berman [38], and Borodin [11]. We believe that results of this form can provide important insights in algorithms and computational complexity, and it would be very interesting to see further advances in this direction.

---

### References

- 1 Scott Aaronson and Avi Wigderson. Algebrization: A new barrier in complexity theory. *Transactions on Computation Theory (TOCT)*, 1(1), 2009.
- 2 Miklós Ajtai.  $\sum_1^1$ -formulae on finite structures. *Annals of Pure and Applied Logic*, 24(1):1–48, 1983.
- 3 Miklós Ajtai and Michael Ben-Or. A theorem on probabilistic constant depth computations. In *Symposium on Theory of Computing (STOC)*, pages 471–474, 1984.

- 4 Eric Allender and Michal Koucký. Amplifying lower bounds by means of self-reducibility. *J. ACM*, 57(3), 2010.
- 5 Noga Alon and Ravi B. Boppana. The monotone circuit complexity of boolean functions. *Combinatorica*, 7(1):1–22, 1987.
- 6 Noga Alon and Joel H. Spencer. *The Probabilistic Method*. Wiley, New York, 1992.
- 7 Alexander E. Andreev. On a method for obtaining lower bounds for the complexity of individual monotone functions. *Soviet Math. Dokl*, 31(3):530–534, 1985.
- 8 Sanjeev Arora and Boaz Barak. *Complexity Theory: A Modern Approach*. Cambridge University Press, Cambridge, 2009.
- 9 Theodore P. Baker, John Gill, and Robert Solovay. Relativizations of the  $P = ?$  NP Question. *SIAM J. Comput.*, 4(4):431–442, 1975.
- 10 Hans L. Bodlaender, Rodney G. Downey, Michael R. Fellows, and Danny Hermelin. On problems without polynomial kernels. *J. Comput. Syst. Sci.*, 75(8):423–434, 2009.
- 11 Allan Borodin. Horner’s rule is uniquely optimal. In *International Symposium on the Theory of Machines and Computations*, pages 45–57, 1971.
- 12 Harry Buhrman and John M. Hitchcock. NP-hard sets are exponentially dense unless  $\text{coNP} \subseteq \text{NP/poly}$ . In *Conference on Computational Complexity (CCC)*, pages 1–7, 2008.
- 13 Arkadev Chattopadhyay and Rahul Santhanam. Lower bounds on interactive compressibility by constant-depth circuits. In *Symposium on Foundations of Computer Science (FOCS)*, pages 619–628, 2012.
- 14 Holger Dell. A simple proof that AND-compression of NP-complete problems is hard. ECCC Report TR14-75, 2014.
- 15 Holger Dell and Dieter van Melkebeek. Satisfiability allows no nontrivial sparsification unless the polynomial-time hierarchy collapses. *J. ACM*, 61(4):23, 2014.
- 16 Andrew Drucker. New limits to classical and quantum instance compression. In *Symposium on Foundations of Computer Science (FOCS)*, pages 609–618, 2012.
- 17 Bella Dubrov and Yuval Ishai. On the randomness complexity of efficient sampling. In *Symposium on Theory of Computing (STOC)*, pages 711–720, 2006.
- 18 Sebastian Faust, Tal Rabin, Leonid Reyzin, Eran Tromer, and Vinod Vaikuntanathan. Protecting circuits from leakage: the computationally-bounded and noisy cases. In *International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*, pages 135–156, 2010.
- 19 William Feller. Generalization of a probability limit theorem of Cramér. *Transactions of the American Mathematical Society*, 54(3):361–372, 1943.
- 20 Jürgen Forster. A linear lower bound on the unbounded error probabilistic communication complexity. *J. Comput. Syst. Sci.*, 65(4):612–625, 2002.
- 21 Jürgen Forster, Matthias Krause, Satyanarayana V. Lokam, Rustam Mubarakzjanov, Niels Schmitt, and Hans-Ulrich Simon. Relations between communication complexity, linear arrangements, and computational complexity. In *Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, pages 171–182, 2001.
- 22 Lance Fortnow and Rahul Santhanam. Infeasibility of instance compression and succinct pcps for NP. *J. Comput. Syst. Sci.*, 77(1):91–106, 2011.
- 23 Merrick L. Furst, James B. Saxe, and Michael Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical Systems Theory*, 17(1):13–27, 1984.
- 24 Parikshit Gopalan and Rocco A. Servedio. Learning and lower bounds for  $AC^0$  with threshold gates. In *International Workshop on Randomization and Computation (RANDOM)*, pages 588–601, 2010.
- 25 András Hajnal, Wolfgang Maass, Pavel Pudlák, Mario Szegedy, and György Turán. Threshold circuits of bounded depth. *J. Comput. Syst. Sci.*, 46(2):129–154, 1993.

- 26 Danny Harnik and Moni Naor. On the compressibility of NP instances and cryptographic applications. *SIAM J. Comput.*, 39(5):1667–1713, 2010.
- 27 Johan Håstad. Almost optimal lower bounds for small depth circuits. In *Symposium on Theory of Computing (STOC)*, pages 6–20, 1986.
- 28 V. M. Khrapchenko. A method of determining lower bounds for the complexity of  $\pi$ -schemes. *Math. Notes Acad. of Sci. (USSR)*, 10(1):474–479, 1971.
- 29 Swastik Kopparty and Srikanth Srinivasan. Certifying polynomials for  $AC^0(\oplus)$  circuits, with applications. In *Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, pages 36–47, 2012.
- 30 Eyal Kushilevitz and Noam Nisan. *Communication Complexity*. Cambridge University Press, 1997.
- 31 Jiří Matoušek and Jan Vondrák. Lecture notes on the probabilistic method, 2008.
- 32 Eduard I. Nečiporuk. On a Boolean function. *Soviet Math. Dokl.*, 7(4):999–1000, 1966.
- 33 Igor C. Oliveira. Algorithms versus circuit lower bounds. ECCC Report TR13-117, 2013.
- 34 Christos H. Papadimitriou and Michael Sipser. Communication complexity. *J. Comput. Syst. Sci.*, 28(2):260–269, 1984.
- 35 Alexander A. Razborov. Lower bounds on monotone complexity of the logical permanent. *Mathematical Notes*, 37(6):485–493, 1985.
- 36 Alexander A. Razborov. Lower bounds on the size of constant-depth networks over a complete basis with logical addition. *Mathematicheskije Zametki*, 41(4):598–607, 1987.
- 37 Alexander A. Razborov and Steven Rudich. Natural proofs. *J. Comput. Syst. Sci.*, 55(1):24–35, 1997.
- 38 Steven Rudich and Leonard Berman. Optimal circuits and transitive automorphism groups. In *International Colloquium on Automata, Languages and Programming (ICALP)*, pages 516–524, 1988.
- 39 Rahul Santhanam. Ironic complicity: Satisfiability algorithms and circuit lower bounds. *Bulletin of the EATCS*, 106:31–52, 2012.
- 40 Roman Smolensky. Algebraic methods in the theory of lower bounds for Boolean circuit complexity. In *Symposium on Theory of Computing (STOC)*, pages 77–82, 1987.
- 41 Srikanth Srinivasan. On improved degree lower bounds for polynomial approximation. In *Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, pages 201–212, 2013.
- 42 Jun Tarui. Smallest formulas for the parity of  $2^k$  variables are essentially unique. *Theor. Comput. Sci.*, 411(26-28):2623–2627, 2010.
- 43 Leslie G. Valiant. Graph-theoretic arguments in low-level complexity. In *Mathematical Foundations of Computer Science (MFCS)*, pages 162–176, 1977.
- 44 Leslie G. Valiant. Exponential lower bounds for restricted monotone circuits. In *Symposium on Theory of Computing (STOC)*, pages 110–117, 1983.
- 45 Emanuele Viola. On the power of small-depth computation. *Foundations and Trends in Theoretical Computer Science*, 5(1):1–72, 2009.
- 46 Ryan Williams. Algorithms for circuits and circuits for algorithms (Invited Talk). In *Conference on Computational Complexity (CCC)*, pages 248–261, 2014.
- 47 Ryan Williams. Faster all-pairs shortest paths via circuit complexity. In *Symposium on Theory of Computing (STOC)*, pages 664–673, 2014.
- 48 Ryan Williams. Nonuniform ACC circuit lower bounds. *J. ACM*, 61(1):2, 2014.
- 49 Andrew Yao. Separating the polynomial-time hierarchy by oracles (preliminary version). In *Symposium on Foundations of Computer Science (FOCS)*, pages 1–10, 1985.

## A Auxiliary results

We use the following standard concentration bound (cf. Alon and Spencer [6], Appendix A).

► **Proposition 1.1.** *Let  $X_1, \dots, X_m$  be independent  $\{0, 1\}$  random variables, where each  $X_i$  is 1 with probability  $p \in [0, 1]$ . In addition, set  $X \stackrel{\text{def}}{=} \sum_i X_i$ , and  $\mu \stackrel{\text{def}}{=} \mathbb{E}[X] = pm$ . Then, for any fixed  $\zeta > 0$ , there exists a constant  $c_\zeta > 0$  such that*

$$\Pr[|X - \mu| > \zeta\mu] < 2e^{-c_\zeta\mu}.$$

## B The degree lower bound in the low-error regime

In this section we describe the proof of the degree lower bound for  $\mathbb{F}_p$ -polynomials approximating  $\text{MOD}_q^n$  in the low error regime. Recall that we use  $\text{MOD}_q^n$  to denote the  $\text{MOD}_q$  function over  $n$  input variables, and that a polynomial  $Q \in \mathbb{F}_p[x_1, \dots, x_n]$   $\varepsilon(n)$ -approximates a Boolean function  $f: \{0, 1\}^n \rightarrow \{0, 1\}$  under the uniform distribution if

$$\Pr_{x \sim \{0, 1\}^n}[Q(x) = f(x)] \geq 1 - \varepsilon(n),$$

where  $x$  is viewed as an element of  $\mathbb{F}_p^n$  or  $\{0, 1\}^n$ , depending on the context.

► **Proposition 2.1** ([36, 40], folklore). *Let  $p, q \geq 2$  be distinct primes. There exist fixed constants  $\delta > 0$  and  $n_0 \in \mathbb{N}$  for which the following holds. For every  $n \geq n_0$  and  $\varepsilon(n) \in [2^{-n}, 1/10q]$ , any polynomial  $P \in \mathbb{F}_p[x_1, \dots, x_n]$  that  $\varepsilon$ -approximates the  $\text{MOD}_q^n$  function with respect to the uniform distribution has degree at least  $\delta \cdot \sqrt{n \cdot \log(1/\varepsilon)}$ .*

The proofs that appear in the literature are concerned with large values of  $\varepsilon$ , and our goal here is to discuss the extension of the degree lower bound to very small  $\varepsilon$ , as stated in Proposition 2.1. For this reason, we will focus on the case where  $q = 2$  and  $p > 2$ , which is slightly simpler. We start with the following lemma.

► **Lemma 2.2.** *For a prime  $p > 2$ , let  $P \in \mathbb{F}_p[x_1, \dots, x_n]$  be a degree- $d$  polynomial that  $\varepsilon(n)$ -approximates  $\text{MOD}_2^n$  over the uniform distribution. Then there exists a polynomial  $Q \in \mathbb{F}_p[y_1, \dots, y_n]$  of degree at most  $d$  and a set  $S \subseteq \{-1, 1\}^n \subseteq \mathbb{F}_p^n$  with  $|S| \geq (1 - \varepsilon)2^n$  such that*

$$\forall y \in S, \quad Q(y) = \prod_{i=1}^n y_i.$$

**Proof.** Let  $T \subseteq \{0, 1\}^n \subseteq \mathbb{F}_p^n$  be a set of size at least  $(1 - \varepsilon)2^n$  such that

$$\forall x \in T, \quad P(x) = \text{MOD}_2^n(x).$$

Consider the map  $\gamma: \{-1, 1\} \rightarrow \{0, 1\}$  computed by the  $\mathbb{F}_p$ -polynomial  $\gamma(y) \stackrel{\text{def}}{=} (1 - y)2^{-1}$ . Observe that  $\gamma(-1) = 1$  and  $\gamma(1) = 0$ . Let  $Q(y_1, \dots, y_n)$  be a polynomial in  $\mathbb{F}_p[y_1, \dots, y_n]$  with  $Q(y) \stackrel{\text{def}}{=} 2P(\gamma(y_1), \dots, \gamma(y_n)) - 1$ , and let

$$S \stackrel{\text{def}}{=} \{y \in \{-1, 1\}^n \mid (y_1, \dots, y_n) = (\gamma^{-1}(x_1), \dots, \gamma^{-1}(x_n)), \text{ where } x \in T\}.$$

Then, using the definition of  $P$ ,  $Q$ ,  $S$ ,  $T$ , and  $\gamma$ , it is not hard to see that

$$\forall y \in S, \quad Q(y) = \prod_{i=1}^n y_i.$$

Finally, observe that  $|S| = |T|$  and  $\deg(Q) \leq \deg(P)$ , which completes the proof of the lemma. ◀

The next lemma shows that polynomials with this property can be very useful when computing functions defined over  $S \subset \mathbb{F}_p^n$ .

► **Lemma 2.3.** *Let  $\mathbb{F}$  be a finite field, and  $a, b \in \mathbb{F}$  be distinct non-zero elements. Assume that  $Q \in \mathbb{F}[x_1, \dots, x_n]$  is a degree- $d$  polynomial, and  $S \subseteq \{a, b\}^n$  is a set such that*

$$\forall x \in S, \quad Q(x) = \prod_{i=1}^n x_i.$$

*Then, for every function  $f: S \rightarrow \mathbb{F}$ , there is a polynomial  $Q_f \in \mathbb{F}[x_1, \dots, x_n]$  with degree at most  $(n + d)/2$  such that*

$$\forall x \in S, \quad Q_f(x) = f(x).$$

**Proof.** Fix a function  $f: S \rightarrow \mathbb{F}$ , and let  $P_f$  be a multilinear polynomial such that, for all  $x \in S$ ,  $P_f(x) = f(x)$ . For instance, since  $a$  and  $b$  are distinct elements of  $\mathbb{F}$ , we can take

$$P_f(x) \stackrel{\text{def}}{=} \sum_{x \in S} f(x) \cdot \left( \prod_{i: x_i=a} (b - x_i)(b - a)^{-1} \right) \left( \prod_{i: x_i=b} (a - x_i)(a - b)^{-1} \right).$$

Now consider any monomial  $M(x) \stackrel{\text{def}}{=} \prod_{i \in I} x_i$ , where  $I \subseteq [n]$ . Since  $a$  and  $b$  are non-zero, for any  $y \in S \subseteq \{a, b\}^n$ , we have

$$\begin{aligned} \prod_{i \in I} y_i &= \left( \prod_{i \in [n]} y_i \right) \left( \prod_{i \notin I} y_i^{-1} \right) \\ &= Q(y) \cdot \left( \prod_{i \notin I} a^{-1}(b - y_i)(b - a)^{-1} + b^{-1}(a - y_i)(a - b)^{-1} \right), \end{aligned}$$

where  $Q$  is the polynomial granted by the statement of the lemma. Therefore, each monomial in  $P_f$  defined over a subset  $I \subseteq [n]$  can be replaced by a monomial of degree at most  $\min(|I|, d + n - |I|) \leq (n + d)/2$ , in the sense that the new polynomial is still correct on every input in  $S$ . Consequently, there exists a polynomial  $Q_f$  for  $f$  with degree at most  $(n + d)/2$ , as claimed by the lemma. ◀

In other words, if  $d$  is small, there exist polynomials of degree much smaller than  $n$  for all functions with domain  $S$  and codomain  $\mathbb{F}$ . This is impossible for large sets  $S$ , via a simple counting argument. In order to formalize this argument and obtain good parameters, we rely on a certain lower bound for the binomial distribution. The next lemma follows from more general results presented in Feller [19]. We follow closely the exposition in Matoušek and Vondrák [31].

► **Lemma 2.4.** *For an even integer  $n \in \mathbb{N}$ , consider independent random variables  $X_1, \dots, X_n$ , where each  $X_i$  attains values 0 and 1, each with probability 1/2. Let  $X \stackrel{\text{def}}{=} \sum_{i \in [n]} X_i$ . Then, for any integer  $t \in [0, n/8]$ ,*

$$\Pr \left[ X \geq \frac{n}{2} + t \right] \geq \frac{1}{15} \cdot e^{-16t^2/n}.$$

**Proof.** For convenience, let  $n = 2m$ . Then,

$$\begin{aligned}
\Pr[X \geq m+t] &= 2^{-2m} \sum_{j=t}^m \binom{2m}{m+j} \\
&\geq 2^{-2m} \sum_{j=t}^{2t-1} \binom{2m}{m+j} \\
&= 2^{-2m} \sum_{j=t}^{2t-1} \binom{2m}{m} \frac{m}{m+j} \cdot \frac{m-1}{m+j-1} \cdots \frac{m-j+1}{m+1} \\
&\geq \frac{1}{2\sqrt{m}} \sum_{j=t}^{2t-1} \prod_{i=1}^j \left(1 - \frac{j}{m+i}\right) \quad (\text{since } \binom{2m}{m} \geq 2^{2m}/(2\sqrt{m})) \\
&\geq \frac{t}{2\sqrt{m}} \left(1 - \frac{2t}{m}\right)^{2t} \\
&\geq \frac{t}{2\sqrt{m}} \cdot e^{-8t^2/m} \quad (\text{since } 1-x \geq e^{-2x} \text{ for } 0 \leq x \leq 1/2).
\end{aligned}$$

The lemma now follows depending on the value of  $t$ . Observe that if  $t \geq \frac{1}{4}\sqrt{m}$  then the last expression is lower bounded by  $\frac{1}{8}e^{-16t^2/n}$ . On the other hand, for  $0 \leq t < \frac{1}{4}\sqrt{m}$ , we get that  $\Pr[X \geq m+t] \geq \Pr[X \geq m + \frac{1}{4}\sqrt{m}] \geq \frac{1}{8}e^{-1/2} \geq \frac{1}{15}$ , which completes the proof.  $\blacktriangleleft$

Finally, we combine these lemmas in order to prove Proposition 2.1 for primes  $q = 2$  and  $p > 2$ .

**Proof.** Let  $P \in \mathbb{F}_p[x_1, \dots, x_n]$  be a degree- $d$  polynomial that  $\varepsilon(n)$ -approximates the  $\text{MOD}_2^n$  function over the uniform distribution. Assume without loss of generality that  $n$  is even, since otherwise we can obtain a polynomial  $Q \in \mathbb{F}_p[x_1, \dots, x_{n+1}]$  with degree at most  $2d$  that  $\varepsilon(n)$ -approximates  $\text{MOD}_2^{n+1}$  with respect to  $\{0, 1\}^{n+1}$  (i.e., apply  $P$  to the first  $n$  variables, then compose with the appropriate function over two input variables).

It follows from Lemmas 2.2 and 2.3 that there exists a set  $S \subseteq \{-1, 1\}^n \subseteq \mathbb{F}_p^n$  of size  $(1 - \varepsilon)2^n$  such that, for every function  $f: S \rightarrow \mathbb{F}_p$ , there exists a polynomial  $Q_f \in \mathbb{F}_p[x_1, \dots, x_n]$  of degree at most  $d' \stackrel{\text{def}}{=} (n+d)/2$  that agrees with  $f$  over  $S$ .

Let  $\mathcal{F}$  be the set of such functions. Clearly,  $|\mathcal{F}| = |\mathbb{F}_p|^{|S|}$ . On the other hand, since  $S \subseteq \{-1, 1\}^n$ , we can assume that each polynomial  $Q_f$  is multilinear. The number of such polynomials with degree at most  $d'$  is upper bounded by  $|\mathbb{F}_p|^M$ , where  $M \stackrel{\text{def}}{=} \sum_{i=0}^{d'} \binom{n}{i}$ . Therefore,  $|\mathbb{F}_p|^{|S|} \leq |\mathcal{F}| \leq |\mathbb{F}_p|^M$ , and we get that

$$\sum_{i=0}^{(n+d)/2} \binom{n}{i} \geq (1 - \varepsilon) \cdot 2^n. \quad (6)$$

We use this inequality to lower bound  $d$  in terms of  $n$  and  $\varepsilon$ . First, Equation 6 can be rewritten as

$$2^{-n} \cdot \sum_{i > (n+d)/2} \binom{n}{i} \leq \varepsilon. \quad (7)$$

On the other hand, it follows from Lemma 2.4 that, for any  $d \in [0, n/8]$ ,

$$\frac{1}{15} \cdot \exp\left(-\frac{16}{n} \cdot \left(\frac{d}{2} + 1\right)^2\right) \leq \Pr\left[X > \frac{n}{2} + \frac{d}{2}\right] = 2^{-n} \cdot \sum_{i > (n+d)/2} \binom{n}{i}. \quad (8)$$

Therefore, we obtain from Equations 7 and 8 that  $d = \Omega(\sqrt{n \cdot \log(1/\varepsilon)})$  for any  $\varepsilon(n) \in [2^{-n}, 1/20]$ , which completes the proof.  $\blacktriangleleft$

## C Improved approximation of $\text{AC}^0[p]$ circuits by polynomials

For convenience of the reader, we describe in this section how to approximate Boolean circuits by bounded-degree polynomials in the low-error regime. We assume the following classic result, obtained in slightly different forms by Razborov [36] and Smolensky [40].

► **Proposition 3.1** ([36], [40]). *Let  $p$  be a fixed prime. There exists a constant  $\beta = \beta(p) \in \mathbb{N}$  such that, for every  $d = d(n) \geq 1$  and  $s = s(n) \geq 1$ , any  $\text{AC}_d^0[p](s(n))$  circuit admits an  $1/(6s)$ -error probabilistic polynomial  $\mathbf{Q}(x_1, \dots, x_n) \in \mathbb{F}_p[x_1, \dots, x_n]$  of degree at most  $(\beta \cdot \log \max\{s, 2\})^d$ .*

We are now ready to describe the proof of the degree upper bound obtained by Kopparty and Srinivasan [29], which allows us to obtain better bounds when the error is sufficiently small.

► **Proposition 3.2** ([29]). *Let  $p$  be a fixed prime. There exists a constant  $\alpha = \alpha(p) \in \mathbb{N}$  such that, for every  $\delta \in (0, 1/2)$  and  $d(n) \geq 2$ , any  $\text{AC}_d^0[p](s(n))$  circuit  $C$  admits a  $\delta$ -error probabilistic polynomial  $\mathbf{Q}(x_1, \dots, x_n) \in \mathbb{F}_p[x_1, \dots, x_n]$  of degree at most  $(\alpha \cdot \log s)^{d-1} \cdot \log(1/\delta)$ . In particular, it follows that for any distribution  $\mathcal{D}$  over  $\{0, 1\}^n$ ,  $C$  is  $\delta$ -approximated with respect to  $\mathcal{D}$  by a polynomial of degree at most  $(\alpha \cdot \log s)^{d-1} \cdot \log(1/\delta)$ .*

**Proof.** Let  $C$  be an  $\text{AC}^0[p]$  circuit of size  $s$  and depth  $d \geq 2$ . Further, let  $g$  be the top gate of  $C$ , and assume that this gate is fed by  $t \leq s$  input wires  $y_1, \dots, y_t$ , where each  $y_j = g_j(x_1, \dots, x_n)$ . Observe that the corresponding Boolean function over inputs  $x_1, \dots, x_n$  at each gate  $g_j$  is computed by a circuit of size at most  $s$  and depth at most  $d - 1$ , while  $g = g(y_1, \dots, y_t)$  is computed by a circuit of size one. Let  $\varepsilon \stackrel{\text{def}}{=} 1/(6s)$ . Then, Proposition 3.1 guarantees the existence of probabilistic polynomials  $\mathbf{Q}_j(x_1, \dots, x_n)$  which compute the corresponding functions  $g_j$  with error at most  $\varepsilon$ , where  $\deg(\mathbf{Q}_j) \leq (\beta \cdot \log s)^{d-1}$ . Similarly, since  $g$  is computed by a single gate, there exists a probabilistic polynomial  $\mathbf{Q}_g(y_1, \dots, y_t)$  that computes  $g$  with error at most  $1/6$ , where  $\deg(\mathbf{Q}_g) \leq \beta$ . By composing these polynomials and applying a union bound, it follows that there exists a probabilistic polynomial  $\mathbf{P}(\vec{x}) \stackrel{\text{def}}{=} \mathbf{Q}_g(\mathbf{Q}_1(\vec{x}), \dots, \mathbf{Q}_t(\vec{x}))$  with  $\deg(\mathbf{P}) \leq (\gamma \cdot \log s)^{d-1}$  that computes  $C$  with error at most  $1/3$ , where  $\gamma = \gamma(p)$  is a fixed constant. Further, by raising this polynomial to  $p - 1$  and applying Fermat's little theorem, we can assume without loss of generality that its output is always Boolean. Since  $d \geq 2$ , the degree becomes at most  $(\gamma' \cdot \log s)^{d-1}$ , where  $\gamma' \leq p \cdot \gamma$ .

Now let  $k = c \cdot \log(1/\delta)$ , for a sufficiently large constant  $c$ . Consider the probabilistic polynomial  $\mathbf{M}(\vec{x}) \stackrel{\text{def}}{=} \mathbf{M}(\mathbf{P}_1(\vec{x}), \dots, \mathbf{P}_k(\vec{x}))$ , where  $\mathbf{M}$  is a degree  $k$  polynomial that computes  $\text{Majority}_k$  exactly, and each  $\mathbf{P}_i$  is an independent copy of  $\mathbf{P}$ . It follows from Proposition 1.1 that  $\mathbf{M}$  is a probabilistic polynomial of degree at most  $(\alpha \cdot \log s)^{d-1} \cdot \log(1/\delta)$  that computes  $C$  with error at most  $\delta$ , where  $\alpha = \alpha(\gamma', c) = \alpha(p)$  is an appropriate constant.  $\blacktriangleleft$



# Lower Bounds for Depth Three Arithmetic Circuits with Small Bottom Fanin

Neeraj Kayal<sup>1</sup> and Chandan Saha<sup>2</sup>

- 1 Microsoft Research India, IN  
neeraka@microsoft.com
- 2 Indian Institute of Science, IN  
chandan@csa.iisc.ernet.in

---

## Abstract

Shpilka and Wigderson [22] had posed the problem of proving exponential lower bounds for (nonhomogeneous) depth three arithmetic circuits with bounded bottom fanin over a field  $\mathbb{F}$  of characteristic zero. We resolve this problem by proving a  $N^{\Omega(\frac{d}{\tau})}$  lower bound for (nonhomogeneous) depth three arithmetic circuits with bottom fanin at most  $\tau$  computing an explicit  $N$ -variate polynomial of degree  $d$  over  $\mathbb{F}$ .

Meanwhile, Nisan and Wigderson [18] had posed the problem of proving superpolynomial lower bounds for homogeneous depth five arithmetic circuits. Over fields of characteristic zero, we show a lower bound of  $N^{\Omega(\sqrt{d})}$  for homogeneous depth five circuits (resp. also for depth three circuits) with bottom fanin at most  $N^\mu$ , for any fixed  $\mu < 1$ . This resolves the problem posed by Nisan and Wigderson only partially because of the added restriction on the bottom fanin (a general homogeneous depth five circuit has bottom fanin at most  $N$ ).

**1998 ACM Subject Classification** F.2.1 Numerical Algorithms and Problems, I.1.1 Expressions and Their Representation

**Keywords and phrases** arithmetic circuits, depth three circuits, lower bound, bottom fanin

**Digital Object Identifier** 10.4230/LIPIcs.CCC.2015.158

## 1 Introduction

The problem of proving super-polynomial lower bounds for arithmetic circuits occupies a central position in algebraic complexity theory, much like the problem of proving super-polynomial lower bounds for Boolean circuits does in Boolean complexity. The model of arithmetic circuits is an algebraic analogue of the model of Boolean circuits: an arithmetic circuit contains addition (+) and multiplication ( $\times$ ) gates and it naturally computes a polynomial in the input variables over some underlying field. We typically allow the input edges to a + gate to be labelled with arbitrary constants from the underlying field  $\mathbb{F}$  so that a + gate can in fact compute an arbitrary  $\mathbb{F}$ -linear combination of its inputs. As a possible stepping stone, researchers have focused on restricted (but still nontrivial and interesting) subclasses of arithmetic circuits. In particular, circuits of low depth<sup>1</sup> are interesting for they correspond to computation which is highly parallel. But despite a lot of attention, proving superpolynomial lower bounds for even bounded depth arithmetic circuits remains an outstanding open problem.

---

<sup>1</sup> Recall that the depth of a circuit is the maximum length of any path in the circuit.





**Notation for low depth circuits.** Bounded depth arithmetic circuits<sup>2</sup> consist of alternating layers of addition and multiplication gates. We will denote an arithmetic circuit of depth  $\Delta$  by a sequence of  $\Delta$  symbols wherein each symbol (either  $\Sigma$  or  $\Pi$ ) denotes the nature of the gates at the corresponding layer and the leftmost symbol indicates the nature of the gates at the output layer. For example, a  $\Sigma\Pi\Sigma$  circuit with input  $\mathbf{x} = (x_1, x_2, \dots, x_n)$  computes a polynomial in the following manner:

$$C(\mathbf{x}) = \sum_i \prod_j \left( a_{ij0} + \sum_{k=1}^n a_{ijk} x_k \right), \quad \text{where each } a_{ijk} \in \mathbb{F}. \quad (1)$$

In dealing with circuits it is useful to keep track of the fanin to various gates. Towards this end, we extend the above notation and allow integer superscripts on the gate symbols (i.e.  $\Sigma$  or  $\Pi$  symbols) which denotes an upper bound on the fanin of any gate in the corresponding layer<sup>3</sup>. So for example a  $\Sigma^{[s]}\Pi^{[e]}\Sigma^{[\tau]}$  circuit computes a polynomial of the form:

$$C(\mathbf{x}) = \sum_{i \leq s} \prod_{j \leq e} \left( \sum_{k \leq \tau} a_{ijk} \cdot y_{ijk} \right) \quad \text{where each } a_{ijk} \in \mathbb{F} \text{ and } y_{ijk} \in \mathbf{x} \cup \{1\}.$$

while a  $\Sigma\Pi^{[a]}\Sigma\Pi^{[b]}$  circuit computes a polynomial in the following manner:

$$C(\mathbf{x}) = \sum_i \prod_{j \leq a} Q_{ij}(\mathbf{x}) \quad \text{where } \deg Q_{ij} \leq b \text{ for all } i \text{ and } j.$$

**Depth Three Circuits.** Being the shallowest nontrivial subclass of arithmetic circuits, depth three arithmetic circuits, also denoted as  $\Sigma\Pi\Sigma$  circuits<sup>4</sup> have been intensely investigated.  $\Sigma\Pi\Sigma$  circuits (more specifically tensors) arise naturally in the investigation of the complexity of polynomial multiplication and matrix multiplication<sup>5</sup>. Moreover, the optimal formula/circuit for some well known families of polynomials are in fact depth three circuits. In particular, the best known circuit for computing the permanent  $\text{Perm}_d$  is known as Ryser’s formula [19] which is a (homogeneous<sup>6</sup>) depth three circuit of size  $O(d^2 \cdot 2^d)$ . Recently it was shown [7] that (nonhomogeneous)  $\Sigma\Pi\Sigma$  circuits are surprisingly powerful – any polynomial  $f$  of *small* circuit complexity can also be computed by a (nonhomogeneous)  $\Sigma\Pi\Sigma$  circuit which

<sup>2</sup> Throughout the rest of this paper, we shall deal with bounded depth circuits – indeed of depth at most 5. In this context, we will often use the words formulas and circuits interchangeably, as depth- $\Delta$  circuits can be converted to depth- $\Delta$  formulas with only a polynomial blow-up in size.

<sup>3</sup> If there is no superscript on the symbol for a layer, then the fanin at that layer is allowed to be arbitrary.

<sup>4</sup> Depth three circuits with a product gate at the output, i.e.  $\Pi\Sigma\Pi$ -circuits, are uninteresting from the perspective of proving lower bounds for they cannot even compute irreducible polynomials of degree more than 1 (regardless of size).

<sup>5</sup> For example it can be shown that the product of two  $n \times n$  matrices can be computed with  $\tilde{O}(n^\omega)$  arithmetic operations if and only if the polynomial

$$M_n = \sum_{i \in [n]} \sum_{j \in [n]} \sum_{k \in [n]} x_{ij} \cdot y_{jk} \cdot z_{ki}$$

can be computed by a  $\Sigma\Pi\Sigma$  circuit where the top fanin  $s$  is at most  $\tilde{O}(n^\omega)$ .

<sup>6</sup> Recall that a multivariate polynomial is said to be homogeneous if all its monomials have the same total degree. An arithmetic circuit is said to be *homogeneous* if the polynomial computed at every internal node of the circuit is a homogeneous polynomial. It is a folklore result (cf. the survey by Shpilka and Yehudayoff [24]) that as far as computation by polynomial-sized arithmetic circuits of unbounded depth is concerned one can assume without loss of generality that the circuit is homogeneous. Specifically, if a homogeneous polynomial  $f$  of degree  $d$  can be computed by an (unbounded depth) arithmetic circuit of size  $s$ , then it can also be computed by a *homogeneous* circuit of size  $O(d^2 \cdot s)$ .

is *not too large*. Specifically<sup>7</sup>, if an  $n$ -variate polynomial  $f$  of degree  $d$  can be computed by  $\text{poly}(n)$ -sized circuits, then it can also be computed by  $n^{O(\sqrt{d})}$ -sized  $\Sigma\Pi\Sigma$  circuit<sup>8</sup>.

**Lower Bounds for  $\Sigma\Pi\Sigma$  circuits.** In a very influential piece of work, Nisan and Wigderson [18] showed that over any field  $\mathbb{F}$ , any *homogeneous*  $\Sigma\Pi\Sigma$  circuit computing the determinant  $\text{Det}_d$  must be of size  $2^{\Omega(d)}$ . Grigoriev and Karpinski [5], and Grigoriev and Razborov [6] showed that any  $\Sigma\Pi\Sigma$  arithmetic circuit over any *fixed* finite field computing  $\text{Det}_d$  must be of size at least  $2^{\Omega(d)}$ . This also implies that any  $\Sigma\Pi\Sigma$  arithmetic circuit *over integers* computing  $\text{Det}_d$  must be of size at least  $2^{\Omega(d)}$ . Raz and Yehudayoff give  $2^{\Omega(d)}$  lower bounds for *multilinear*  $\Sigma\Pi\Sigma$  circuits<sup>9</sup>. But despite all this progress, even a superpolynomial lower bound for unrestricted  $\Sigma\Pi\Sigma$  circuits (over an infinite field) has remained elusive. The best known lower bound in the general  $\Sigma\Pi\Sigma$  case is the quadratic lower bound due to Shpilka and Wigderson [22]. For more on  $\Sigma\Pi\Sigma$  circuits, we refer the reader to the thesis of Shpilka [21] and the references therein.

**$\Sigma\Pi\Sigma$  circuits with small bottom fanin.** Nisan and Wigderson noted that (nonhomogeneous)  $\Sigma\Pi\Sigma$  circuits with bottom fanin just two can be exponentially more powerful than homogeneous  $\Sigma\Pi\Sigma$  circuits – any homogeneous  $\Sigma\Pi\Sigma$  circuit computing the elementary symmetric polynomial of degree  $n$  on  $2n$  variables<sup>10</sup> must be of size  $2^{\Omega(n)}$  but it can be computed by just  $O(n^2)$ -sized  $\Sigma\Pi\Sigma^{[2]}$  circuits<sup>11</sup>. They also noted that this contrasts sharply with the the exponential lower bounds for MAJORITY in the Boolean model and over fixed finite fields. Recently, Ramprasad Saptharishi [20] pointed out to us that the depth reduction in [7] actually yields  $\Sigma\Pi\Sigma^{[O(\sqrt{d})]}$ -circuits. This indicates that (nonhomogeneous)  $\Sigma\Pi\Sigma^{[\tau]}$ -circuits are interesting and motivates the effort to prove lower bounds for them. Indeed, Shpilka and Wigderson [22] had already noted this frontier in arithmetic complexity and explicitly posed the problem of proving lower bounds for (nonhomogeneous) depth three circuits with bounded bottom fanin (over fields of characteristic zero). We resolve this challenge here by proving exponential lower bounds for such circuits. Our proof techniques are based on recent developments in arithmetic circuit lower bounds.

**Recent lower bound results.** A series of recent works have built upon the work of Nisan and Wigderson [18] to prove lower bounds for *homogeneous* depth four circuits. Motivated

<sup>7</sup> The quantitative version mentioned here is due to an improvement by Tavenas [25].

<sup>8</sup> This depth reduction is only valid over fields of characteristic zero.

<sup>9</sup> The results of Raz and Yehudayoff are more general and extend to lower bounds for any constant depth multilinear circuit.

<sup>10</sup> The elementary polynomial of degree  $n$  on  $2n$  formal variables is the arithmetic analog of the MAJORITY function. Formally, it is defined as

$$\text{ESYM}_n(x_1, \dots, x_{2n}) \stackrel{\text{def}}{=} \sum_{\substack{S \subseteq [2n] \\ |S|=n}} \prod_{i \in S} x_i.$$

<sup>11</sup> More accurately, [18] attribute Michael Ben-Or for an  $O(n^2)$ -sized  $\Sigma\Pi\Sigma$  circuit for  $\text{ESYM}_n(x_1, x_2, \dots, x_{2n})$  which has the following specific form:

$$\text{ESYM}_n(\mathbf{x}) = \sum_{i=1}^{2n+1} a_i \prod_{j=1}^{2n} (x_j + i),$$

where the  $a_i$ 's are appropriate field constants.

by the depth reduction results of Agrawal and Vinay [1] and Koiran [14] and Tavenas [25] and using a complexity measure introduced in Kayal [10], the work of Gupta, Kamath, Kayal and Saptharishi [8] and Kayal, Saha and Saptharishi [13] have led to lower bounds of  $n^{\Omega(\sqrt{d})}$  for homogeneous depth four circuits of bottom fanin  $O(\sqrt{d})$ . Follow-up work by Fournier, Limaye, Malod and Srinivasan [4] showed the same lower bound for a family of polynomials in VP. Subsequently, work by Kayal, Limaye, Saha and Srinivasan [12, 11] removed the restriction on the bottom fanin and obtained a  $n^{\Omega(\sqrt{d})}$  lower bound for homogeneous depth four circuits for a family of polynomials in VNP<sup>12</sup>. Follow-up work by Kumar and Saraf [15] showed the same lower bounds for a family of polynomials in VP<sup>13</sup>.

**Our results.** Our first result is a lower bound of  $N^{\Omega(\frac{d}{\tau})}$  for (nonhomogeneous)  $\Sigma\Pi\Sigma^{[\tau]}$  circuits which resolves an open problem (specifically, Problem 7.5 in [23]) posed by Shpilka and Wigderson in [22]. It also implies that the depth reduction result of [7] is optimal *assuming that the resulting depth three circuit has bottom fanin at most  $O(\sqrt{d})$* . The formal statement is as follows.

► **Theorem 1.1** (Lower Bound for  $\Sigma\Pi\Sigma^{[\tau]}$  circuits). *Let  $\mathbb{F}$  be a field of characteristic zero. There is a family of  $N$ -variate, degree  $d$  polynomials  $\{f_N\}$  in VP with  $N = d^{O(1)}$  such that any  $\Sigma\Pi\Sigma^{[\tau]}$  circuit over  $\mathbb{F}$  computing  $f_N$  must have top fanin at least  $N^{\Omega(\frac{d}{\tau})}$ .*

We would like to stress here that there is no restriction of homogeneity on the  $\Sigma\Pi\Sigma^{[\tau]}$  formula in the above statement. Indeed the formal degree of the  $\Sigma\Pi\Sigma^{[\tau]}$  circuit can be arbitrarily large (say doubly exponential) and yet we obtain the stated lower bound on the top fanin. We prove Theorem 1.1 by first showing a reduction from  $\Sigma\Pi\Sigma^{[\tau]}$  circuits to a subclass of homogeneous  $\Sigma\Pi\Sigma\Pi\Sigma^{[\tau]}$  circuits<sup>14</sup> (using a result implicit in [22] and [7]; see Lemma 4.1 in Section 4). It turns out fortunately that the proof techniques/complexity measure used in [11, 15] are readily applicable to this subclass of homogeneous  $\Sigma\Pi\Sigma\Pi\Sigma^{[\tau]}$  circuits and this yields the above lower bound. Having obtained a lower bound for a subclass of homogeneous  $\Sigma\Pi\Sigma\Pi\Sigma$  circuits, can our techniques be pushed further to yield lower bounds for general homogeneous  $\Sigma\Pi\Sigma\Pi\Sigma$  formulas? It turns out that proving superpolynomial lower bounds for general homogeneous  $\Sigma\Pi\Sigma\Pi\Sigma$  formulas was explicitly posed as an open problem by Nisan and Wigderson in [18]. We next give a lower bound for homogeneous  $\Sigma\Pi\Sigma\Pi\Sigma$  formulas with small bottom fanin. It resolves the above problem only partially because of the added restriction on the bottom fanin.

► **Theorem 1.2** (Lower Bound for homogeneous  $\Sigma\Pi\Sigma\Pi\Sigma^{[\tau]}$  circuits). *Let  $\mathbb{F}$  be a field of characteristic zero and  $\mu \in [0, 1)$  be any fixed positive real number less than 1. Let  $\alpha = \frac{2\mu+1}{1-\mu}$  and  $\tau = O(N^\mu)$ . There is a family of  $N$ -variate, degree  $d$  polynomials  $\{f_N\}$  in VNP with  $N \in [d^{2+\alpha}, 2d^{2+\alpha}]$  such that any homogeneous  $\Sigma\Pi\Sigma\Pi\Sigma^{[\tau]}$  formula over  $\mathbb{F}$  computing  $f_N$  has size  $N^{\Omega(\sqrt{d})}$ .*

The family of polynomials in the above theorem is the Nisan-Wigderson design based polynomials introduced in [13], and later used in [11, 15], but with an altered set of parameters.

<sup>12</sup> Meanwhile, an independent work by Kumar and Saraf [16] also showed a  $n^{\Omega(\log \log n)}$  lower bound for general homogeneous depth-4 circuits without the bottom fanin restriction.

<sup>13</sup> The result of [15] is also valid over any field  $\mathbb{F}$ .

<sup>14</sup> The reduction from  $\Sigma\Pi\Sigma$  formulas to homogeneous  $\Sigma\Pi\Sigma\Pi\Sigma$  formulas yields a restricted class of homogeneous  $\Sigma\Pi\Sigma\Pi\Sigma$  formulas wherein every product gate in the layer closest to the input layer is actually an *exponentiation gate*, i.e. a product gate all of whose inputs originate from the source node  $g$ , so that its output is of the form  $g^e$  for some  $e \in \mathbb{Z}_{\geq 1}$ . We denote such formulas as  $\Sigma\Pi\Sigma\wedge\Sigma$  formulas.

The complexity measure that we use for this result is (almost) the same as the one introduced in [11] called *the dimension of projected shifted partials under random restrictions*. An appropriate adaption of the techniques yields a lower bound for  $N$ -input homogeneous  $\Sigma\Pi\Sigma\Pi\Sigma^{[N^\mu]}$ -circuits for some fixed value of  $\mu < 0.1$ . We felt that it would be worthwhile to push the analysis further and obtain as good a lower bound as possible while allowing the bottom fanin to be as large as possible – specifically, to allow the bottom fanin to be  $N^\mu$  for any constant  $\mu$  that is arbitrarily close to 1. For this, we delve deeper into the analysis of [11] and carefully tune it at certain places, including the complexity analysis of the explicit polynomial family for which the lower bound is shown. As a corollary, we also obtain a similar lower bound for (nonhomogeneous)  $\Sigma\Pi\Sigma^{[N^\mu]}$  circuits for any constant  $\mu < 1$ .

► **Corollary 1.3.** *Let  $\mathbb{F}$  be a field of characteristic zero and  $\mu \in [0, 1)$  be any fixed positive real number less than 1. Let  $\alpha = \frac{2\mu+1}{1-\mu}$ . There is a family of  $N$ -variate, degree  $d$  polynomials  $\{f_N\}$  in VNP with  $N \in [d^{2+\alpha}, 2d^{2+\alpha}]$  such that any  $\Sigma\Pi\Sigma^{[N^\mu]}$  formula over  $\mathbb{F}$  computing  $f_N$  has size at least  $N^{\Omega(\sqrt{d})}$ .*

## 2 Proof Overview

**From depth three to homogeneous depth-5.** Let  $f(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$  be a homogeneous  $N$ -variate polynomial of degree  $d$ . It was already observed by Shpilka and Wigderson [22] that if  $f$  is computed by a small (of size  $N^{o(\sqrt{d})}$ )  $\Sigma\Pi\Sigma$  circuit  $C(\mathbf{x})$  then it is also computed by a small (of size  $N^{o(\sqrt{d})}$ ) formula  $D(\mathbf{x})$  which is structurally in a subclass of *homogeneous*  $\Sigma\Pi\Sigma\Pi\Sigma$  formulas. We observe that this reduction from depth three to homogeneous depth-5 preserves the bound on the bottom fanin of the formulas, i.e. if the bottom fanin of  $C(\mathbf{x})$  is bounded by  $\tau$  then same is true for  $D(\mathbf{x})$  (see Lemma 4.1 in Section 4). It turns out that the proof techniques/complexity measure employed in [11, 15] are readily applicable to this subclass of homogeneous  $\Sigma\Pi\Sigma\Pi\Sigma^{[\tau]}$  circuits and this yields the lower bound of theorem 1.1. We then consider general homogeneous  $\Sigma\Pi\Sigma\Pi\Sigma^{[\tau]}$  circuits.

**Homogeneous depth five formulas.** A homogeneous depth-5 formula is a representation of the form

$$D(\mathbf{x}) = \sum_i \prod_j \sum_r Q_{ijr}, \quad (2)$$

where  $Q_{ijr}$  is a product of linear forms. Also, suppose the number of variables in every linear form in  $Q_{ijr}$  (for every  $i, j$  and  $r$ ) is bounded by  $\tau = N^\mu$  for some fixed constant  $\mu < 1$ . To prove a lower bound on the size of  $D(\mathbf{x})$ , our overall strategy is based on the complexity measure introduced in [11] called *the dimension of projected shifted partials under random restrictions*. As is common to many lower bounds, the proof is in two steps:

1. Upper bound the measure for any  $\Sigma\Pi\Sigma\Pi\Sigma^{[\tau]}$ -formula  $D(\mathbf{x})$  as in equation (2), and
2. Lower bound the measure for an explicit (family of) polynomial(s)  $f$ .

Overall, the lower bound follows by comparing these two bounds. We will now describe the complexity measure used and then indicate why it is small for  $\Sigma\Pi\Sigma\Pi\Sigma^{[\tau]}$ -formulas.

**Random restriction.** The random restriction we use in this paper is quite natural and (almost) same as in [11]. We consider the identity (2) and in that set each variable to zero independently at random with probability  $(1 - p)$ , where  $p = d^{-\beta}$  for a suitable constant  $\beta > 0$  (a variable is left untouched with probability  $p$ .) For ease of exposition, it is convenient

to denote a restriction in which a subset of variables  $R \subseteq [N]$  is<sup>15</sup> set to zero (and the variables outside  $R$  are left untouched) as a homomorphism,  $\sigma_R : \mathbb{F}[\mathbf{x}] \mapsto \mathbb{F}[\mathbf{x}]$ . Formally,  $\sigma_R : \mathbb{F}[\mathbf{x}] \mapsto \mathbb{F}[\mathbf{x}]$  is a homomorphism such that  $\sigma_R(f) \stackrel{\text{def}}{=} f|_{x_i=0 \ \forall i \in R}$ . In this notation, a random restriction can also be viewed as constructing an  $R$  by picking every variable independently at random with probability  $1 - p$  and then applying<sup>16</sup> the map  $\sigma_R$  to the expression given by equation (2).

**The complexity measure.** Let  $m = x_{i_1} \cdots x_{i_k}$  be a monomial in  $\mathbf{x}$ . Denote  $\frac{\partial^k}{\partial x_{i_1} \cdots \partial x_{i_k}} f$  by  $\partial_m f$  and define

$$\partial_{\text{ML}}^k f := \{\partial_m f \mid m \text{ is a multilinear monomial of degree } k\}$$

We will refer to  $\partial_{\text{ML}}^k f$  as the set of all *multilinear*  $k$ -th order partial derivatives of  $f \in \mathbb{F}[\mathbf{x}]$ . Let  $\mathbf{x}^{\ell}$  be the set of all multilinear monomials in  $\mathbf{x}$  of degree equal to  $\ell$ . We denote by  $\mathbf{x}^{\ell} \cdot \partial_{\text{ML}}^k f$  the set of all polynomials of the form  $m \cdot g$  where  $m \in \mathbf{x}^{\ell}$  and  $g \in \partial_{\text{ML}}^k f$ . Define a map  $\pi : \mathbb{F}[\mathbf{x}] \mapsto \mathbb{F}[\mathbf{x}]$  such that when  $\pi$  acts on a polynomial  $f$ , it retains only and exactly the *multilinear monomials* of  $f$ . More precisely, let  $M_f$  be the set of all monomials with nonzero coefficients in  $f$ . Then,  $\pi(f) := \sum_u c_u m_u$  where  $m_u$  is a multilinear monomial in  $M_f$  and coefficient of  $m_u$  in  $f$  is  $c_u$ . Naturally,  $\pi$  is a linear map, i.e.  $\pi(af + bg) = a \cdot \pi(f) + b \cdot \pi(g)$  for every  $a, b \in \mathbb{F}$  and  $f, g \in \mathbb{F}[\mathbf{x}]$ . The definition of  $\pi$  extends naturally to sets of polynomials: For  $A \subseteq \mathbb{F}[\mathbf{x}]$ , let  $\pi(A) := \{\pi(f) \mid f \in A\}$ . For integers  $k$  and  $\ell$ , the space of projected shifted partials of  $f$  is the linear span (i.e.  $\mathbb{F}$ -span) of the polynomials in  $\pi(\mathbf{x}^{\ell} \cdot \partial_{\text{ML}}^k f)$ . The measure we use is the dimension of this space of projected shifted partials, denoted by  $\text{DPSP}_{k,\ell}$  (or simply DPSP assuming parameters  $k$  and  $\ell$  are fixed suitably):

$$\text{DPSP}_{k,\ell}(f) := \dim(\pi(\mathbf{x}^{\ell} \cdot \partial_{\text{ML}}^k f)).$$

Observe that the measure  $\text{DPSP}_{k,\ell}$  obeys subadditivity, i.e.  $\text{DPSP}_{k,\ell}(f + g) \leq \text{DPSP}_{k,\ell}(f) + \text{DPSP}_{k,\ell}(g)$ .

**From depth-5 to depth-4.** Let  $D(\mathbf{x})$  be a homogeneous- $\Sigma\Pi\Sigma\Pi\Sigma^{[N^\mu]}$  formula as in equation (2) of size at most  $N^{o(\sqrt{d})}$  so that in particular the total number of  $Q_{ijr}$ 's appearing in it is at most  $s = N^{o(\sqrt{d})}$ . We show that when a random restriction  $\sigma_R$  is applied on  $D(\mathbf{x})$ , then with high probability  $\sigma_R(D(\mathbf{x}))$  can be expressed as  $D_1(\mathbf{x}) + D_2(\mathbf{x})$ , where  $D_1(\mathbf{x})$  is computed by a homogeneous  $\Sigma\Pi\Sigma\Pi^{[\sqrt{d}]}$  formula of top fanin at most  $N^{o(\sqrt{d})}$  and  $D_2(\mathbf{x})$  is a polynomial such that  $\text{DPSP}(D_2(\mathbf{x})) = 0$ . We will argue this shortly but assuming that this happens, we can infer (via subadditivity) that

$$\begin{aligned} \text{DPSP}(\sigma_R(D(\mathbf{x}))) &\leq \text{DPSP}(D_1(\mathbf{x})) + \text{DPSP}(D_2(\mathbf{x})) \\ &= \text{DPSP}(D_1(\mathbf{x})). \end{aligned}$$

$\text{DPSP}(D_1(\mathbf{x}))$  can then be upper bounded using known arguments from [11] which in turn yields an upper bound for  $\text{DPSP}(\sigma_R(D(\mathbf{x})))$ .

<sup>15</sup>  $[N]$  denotes the set of the first  $N$  positive integers, i.e.  $\{1, 2, \dots, N\}$ .

<sup>16</sup> We will use the random restriction in two phases in Section 6 to obtain an appropriate upper bound on the measure for homogeneous  $\Sigma\Pi\Sigma\Pi\Sigma^{[\tau]}$  formulas.

**Using random restrictions to obtain a decomposition.** The reason  $\sigma_R(D(\mathbf{x}))$  decomposes into  $D_1(\mathbf{x})$  and  $D_2(\mathbf{x})$  with high probability is as follows. Let  $t = \sqrt{d}$ . In equation (2), suppose a  $Q_{ijr}$  has degree greater than  $2t$ . Such a  $Q_{ijr}$  can be expressed as  $\tilde{Q}_{ijr} \cdot P_{ijr}$  with  $\deg(\tilde{Q}_{ijr}) = 2t$ , by simply multiplying out  $2t$  linear forms in  $Q_{ijr}$ . Since bottom fanin of  $D(\mathbf{x})$  is bounded by  $N^\mu$ , the number of monomials in  $\tilde{Q}_{ijr}$  is bounded by  $N^{2\mu t}$ . Monomials of  $\tilde{Q}_{ijr}$  are of two kinds – those with individual degree of variables bounded by 2 (and hence have support at least  $t$ ), and those with at least one variable having degree 3 or more. The probability any of the monomials in  $\tilde{Q}_{ijr}$  survives under the action of the random restriction  $\sigma_R$  is less than  $p^t \cdot N^{2\mu t}$ . Running over all  $Q_{ijr}$ , with probability at least  $1 - s \cdot p^t \cdot N^{2\mu t}$ , we have

$$\sigma_R(D(\mathbf{x})) = \sum_i \prod_j \sum_{\deg(Q_{ijr}) \leq 2t} \sigma_R(Q_{ijr}) + P(\mathbf{x}),$$

where every monomial in  $P(\mathbf{x})$  has a variable with degree 3 or more. Now observe that for any multilinear monomial  $m$ , every monomial in  $\partial_m P$  has a variable of degree 2 or more and hence  $\pi(\partial_m P) = 0$ , implying  $\text{DPSP}(P) = 0$ . By taking  $D_1(\mathbf{x}) = \sum_i \prod_j \sum_{r, \deg(Q_{ijr}) \leq 2t} \sigma_R(Q_{ijr})$  and  $D_2(\mathbf{x}) = P(\mathbf{x})$ , we come to the desired conclusion, if the “bad” probability, namely  $s \cdot p^t \cdot N^{2\mu t}$ , is small. Now suppose  $N = d^3$  (as is the case in [11]). Then the bad probability is  $s \cdot N^{-(\frac{\beta}{3} - 2\mu)t}$  which is negligible for any constant  $\mu$  less than  $\beta/6$ . This gives the required decomposition.

**Extension for arbitrary  $\mu < 1$ .** Combining the above decomposition argument with the lower bound available for homogeneous- $\Sigma\Pi\Sigma\Pi^{[\sqrt{d}]}$ -circuits (which imposes some additional constraints on how large  $\beta$  can be), we get that if  $\mu$  is sufficiently small (say, 0.01), any homogeneous  $\Sigma\Pi\Sigma\Pi^{[N^\mu]}$  formula computing the same family of Nisan-Wigderson design based polynomials as used in [11], has size  $N^{\Omega(\sqrt{d})}$ . However, in order to prove the same size lower bound for *any* constant  $\mu < 1$ , we delve deeper into the analysis of [11] and carefully tune it at certain places, including the complexity analysis of the explicit polynomial family for which the lower bound is shown.

### 3 Preliminaries

**Affine forms and linear forms.** An *affine form* is simply another name for a degree one polynomial, with a (possibly) nonzero constant term. Thus an affine form  $\ell(\mathbf{x})$  looks like

$$\ell(\mathbf{x}) = a_0 + a_1x_1 + a_2x_2 + \dots + a_nx_n,$$

where each  $a_i \in \mathbb{F}$ . The *weight* of such an affine form  $\ell(\mathbf{x})$  will be the number of nonzero coefficients in it, i.e.

$$\text{weight of } \ell \stackrel{\text{def}}{=} |\{i \in [0..n] : a_i \neq 0\}|$$

A homogeneous degree one polynomial (i.e. one whose constant term  $a_0$  is zero) we will refer to as a *linear form*.

**Notation for circuits with exponentiation gates.** Sometimes a multiplication gate in our circuit will have the feature that all its incoming edges originate from a single gate  $g$  (thus computing  $g^e$ , if there are  $e$  wires entering the multiplication gate). We will refer to such

gates as *exponentiation gates* and denote them by the symbol  $\wedge$ . So for example, a  $\Sigma\wedge\Sigma$  circuit computes a polynomial in the following manner:

$$C(\mathbf{x}) = \sum_{i \in [s]} \ell_i(\mathbf{x})^{e_i} \quad \text{where each } \ell_i \in \mathbb{F}[\mathbf{x}] \text{ is an affine form.}$$

**A numerical estimate.** The following numerical estimate from [8] will be useful.

► **Lemma 3.1.** *Let  $a(n), f(n), g(n): \mathbb{Z}_{>0} \mapsto \mathbb{Z}$  be integer valued functions such that  $(|f| + |g|) = o(a)$ . Then*

$$\ln \frac{(a + f)!}{(a - g)!} = (f + g) \ln a \pm O\left(\frac{f^2 + g^2}{a}\right)$$

#### 4 Depth Three Circuits with small bottom fanin

In this section, we will first see a reduction from (nonhomogeneous)  $\Sigma\Pi\Sigma^{[\tau]}$  to a subclass of homogeneous  $\Sigma\Pi\Sigma\Pi\Sigma^{[\tau]}$  circuits. It can be easily inferred from the proofs of theorem 5.2 in [23] and lemma V.3<sup>17</sup> in [7] but we nevertheless give a proof here for completeness.

► **Lemma 4.1.** *(implicit in [22] and [7].) Let  $d \geq 1$  be an integer and  $\mathbb{F}$  be an infinite field of characteristic larger than  $d$  (or of zero characteristic). Let  $f(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$  be a homogeneous  $N$ -variate polynomial of degree  $d$  computed by a  $\Sigma^{[s]}\Pi^{[e]}\Sigma^{[\tau]}$  circuit. Then  $f$  can also be computed by a homogeneous  $\Sigma^{[s \cdot \exp(\sqrt{d})]}\Pi\Sigma^{[e]}\wedge\Sigma^{[\tau]}$  circuit.*

**Proof.** The premise that  $f$  can be computed by a  $\Sigma^{[s]}\Pi^{[e]}\Sigma^{[\tau]}$  circuit means that there exist  $s \cdot e$  affine forms  $\ell_{ij}$ 's each of weight at most  $\tau$  such that

$$f(\mathbf{x}) = \sum_{i=1}^s \prod_{j=1}^e \ell_{ij}(\mathbf{x}). \tag{3}$$

**Expressing  $f$  as a sum of projections of elementary symmetric polynomials.** We will first ensure that each of the affine forms  $\ell_{ij}$  has a nonzero constant term. We can do this by applying a random shift of the form  $\mathbf{x} \mapsto \mathbf{x} + \mathbf{a}$  to the above identity. That is, pick a random point  $\mathbf{a} \in \mathbb{F}^n$  and replacing  $\mathbf{x}$  by  $\mathbf{x} + \mathbf{a}$  in the identity (3) we get

$$\begin{aligned} f(\mathbf{x} + \mathbf{a}) &= \sum_{i=1}^s \prod_{j=1}^e \ell_{ij}(\mathbf{x} + \mathbf{a}) \\ &= \sum_{i=1}^s \alpha_i \prod_{j=1}^e (1 + m_{ij}(\mathbf{x})), \text{ where } m_{ij}(\mathbf{x}) \stackrel{\text{def}}{=} \ell_{ij}(\mathbf{x}) - \ell_{ij}(\mathbf{0}) \text{ is a linear form of} \\ &\quad \text{weight at most } \tau \text{ and } \alpha_i \stackrel{\text{def}}{=} \prod_{j=1}^e \ell_{ij}(\mathbf{a}) \end{aligned}$$

Comparing the homogeneous components of degree  $d$  on the two sides of the above identity we get

$$f(\mathbf{x}) = \sum_{i=1}^s \alpha_i \cdot \text{ESYM}_d(m_{i1}, \dots, m_{ie}), \tag{4}$$

<sup>17</sup>Ramprasad Saptharishi [20] has recently communicated to us that the consequence in the original lemma in [7] can be slightly improved quantitatively.



where

$$\text{ESYM}_d(y_1, \dots, y_e) \stackrel{\text{def}}{=} \sum_{\substack{S \subseteq [e] \\ |S|=d}} \prod_{i \in S} y_i$$

is the elementary symmetric polynomial of degree  $d$  on the  $e$  formal variables  $y_1, y_2, \dots, y_e$ .

**Expressing  $\text{ESYM}_d$  in terms of the power symmetric polynomials.** We now use Newton's identities to express each elementary symmetric polynomial that occurs above in terms of the power-symmetric polynomials defined as:

$$\text{PSYM}_r(y_1, \dots, y_e) \stackrel{\text{def}}{=} \sum_{j \in [e]} y_j^r.$$

We use the following implication of Newton's identities (cf. [17]):

$$\text{ESYM}_d = \frac{1}{d!} \cdot \begin{vmatrix} \text{PSYM}_1 & 1 & 0 & 0 & \cdots & 0 & 0 \\ \text{PSYM}_2 & \text{PSYM}_1 & 2 & 0 & \cdots & 0 & 0 \\ \text{PSYM}_3 & \text{PSYM}_2 & \text{PSYM}_1 & 3 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \text{PSYM}_{d-1} & \text{PSYM}_{d-2} & \text{PSYM}_{d-3} & \text{PSYM}_{d-4} & \cdots & \text{PSYM}_1 & d-1 \\ \text{PSYM}_d & \text{PSYM}_{d-1} & \text{PSYM}_{d-2} & \text{PSYM}_{d-3} & \cdots & \text{PSYM}_2 & \text{PSYM}_1 \end{vmatrix}.$$

In particular, this means that  $\text{ESYM}_d$  can be expressed as a polynomial function of the  $\text{PSYM}_i$ 's. Let us now count how many terms are there in such a polynomial expression. Expanding out the determinant above we see that there exist scalars  $\beta_{\mathbf{a}}$ 's such that

$$\text{ESYM}_d(\mathbf{y}) = \sum_{\substack{\mathbf{a}=(a_1, \dots, a_d) \in \mathbb{Z}_{\geq 0}^d \\ \sum_i i \cdot a_i = d}} \beta_{\mathbf{a}} \cdot \prod_{i \in [d]} \text{PSYM}_i^{a_i}(\mathbf{y}). \quad (5)$$

The number of solutions of  $\sum_{i \in [d]} i \cdot a_i = d$  is exactly the number of ways to partition the natural number  $d$  and hence is  $2^{\Theta(\sqrt{d})}$  by the Hardy-Ramanujan estimate for the partition function [9]. Hence the number of terms in the above summation is  $2^{\Theta(\sqrt{d})}$ . In particular this means that  $\text{ESYM}_d(\mathbf{y})$  is computed by a *homogeneous*  $\Sigma^{\lceil \exp(\sqrt{d}) \rceil} \Pi \Sigma^{[e]} \wedge$ -circuit.

**Combining (4) and (5) to get a homogeneous  $\Sigma \Pi \Sigma \wedge \Sigma$  circuit for  $f$ .** If we now replace each occurrence of  $\text{ESYM}_d$  in equation (4) by its homogeneous  $\Sigma \Pi \Sigma \wedge$  circuit given by the identity (5), we see that  $f(\mathbf{x})$  is computed by a homogeneous  $\Sigma^{\lceil s \cdot \exp(\sqrt{d}) \rceil} \Pi \Sigma^{[e]} \wedge \Sigma^{\lceil \tau \rceil}$  circuit. This proves the lemma. ◀

We next observe that the homogeneous  $\Sigma \Pi \Sigma \wedge \Sigma$ -circuit in the outcome of the above lemma corresponds to a certain structured form for expressing  $f$  that we make precise below. For ease of subsequent exposition, let us introduce the following notation/terminology. Let  $m = x_1^{e_1} \cdot x_2^{e_2} \cdot \dots \cdot x_N^{e_N}$  in  $\mathbb{F}[x_1, x_2, \dots, x_N]$  be a monomial. The support of  $m$ , denoted  $\text{Supp}(m)$  is the subset of variables appearing in it, i.e.

$$\text{Supp}(m) \stackrel{\text{def}}{=} \{i : e_i \geq 1\} \subseteq [N].$$

The support size of a polynomial  $Q$ , denoted  $|\text{Supp}(Q)|$  is the maximum support size of any monomial appearing in  $Q$ .



► **Proposition 4.2.** *Let  $d \geq 1$  be an integer and  $\mathbb{F}$  be an infinite field of characteristic larger than  $d$  (or of zero characteristic). Let  $f(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$  be a homogeneous  $N$ -variate polynomial of degree  $d$  computed by a  $\Sigma^{[s]}\Pi^{[e]}\Sigma^{[\tau]}$  circuit. Then  $f$  admits an expression of the form*

$$f(\mathbf{x}) = \sum_i^{s \cdot \exp(\sqrt{d})} \prod_j Q_{ij}, \quad \text{Supp}(Q_{ij}) \leq \tau \tag{6}$$

**Proof.** The premise that  $f$  can be computed by a  $\Sigma^{[s]}\Pi^{[e]}\Sigma^{[\tau]}$  circuit means that there exist  $s \cdot e$  affine forms  $\ell_{ij}$ 's each having at most  $\tau$  nonzero coefficients such that

$$f(\mathbf{x}) = \sum_{i=1}^s \prod_{j=1}^e \ell_{ij}(\mathbf{x}). \tag{7}$$

First observe that if we have a linear form  $\ell$  in which at most  $\tau$  coefficients are nonzero, then for all  $j \geq 1$ , we have

$$\text{Supp}(\ell^j) \leq \tau.$$

In particular, this means that for all  $r \geq 1$  and all  $i \leq s$  we have  $\text{Supp}(\text{PSYM}_r(\ell_{i1}, \ell_{i2}, \dots, \ell_{ie})) \leq \tau$ . By the proof of lemma 4.1 we get that  $f$  can be expressed as a sum of product of the  $\text{PSYM}_r$ 's in a homogeneous fashion, with the expression having  $s \cdot \exp(\sqrt{d})$  many terms. Hence  $f$  has a representation of the form given by equation (6). ◀

This means that our problem reduces to proving lower bounds for representations of the form given by the right-hand side of equation (6) which we refer to as  $\tau$ -supported homogeneous  $\Sigma\Pi\Sigma\Pi$  circuits. It turns out that such representations occur also as an intermediate step in prior work and [11] explicitly gives an  $N^{\Omega(\frac{d}{\tau})}$  lower bound for such representations.

► **Theorem 4.3.** [11]. *There exists an explicit family  $\{f_N\}$  of homogeneous degree  $d$  polynomials on  $N = d^3$  variables in VNP such that any  $\tau$ -supported homogeneous  $\Sigma\Pi\Sigma\Pi$  circuit computing  $f_N$  has top fanin at least  $N^{\Omega(\frac{d}{\tau})}$ .*

► **Remark.** We would like to stress here that the above theorem holds for any  $\tau \geq 1$ . In [11], the analysis was done by setting the parameter  $\ell$  of the measure  $\text{DPSP}_{k,\ell}$  as  $\ell = \frac{N}{2} \left(1 - \frac{k \ln d}{d}\right)$  (where  $k = \frac{\delta d}{\tau}$  for a suitable constant  $\delta > 0$ ). With this choice of  $\ell$ , the parameter  $\tau$  has to be  $\Omega(\ln d)$  or else  $\ell$  becomes negative (which does not quite make sense). We note here that the choice of  $\ell$  can be altered (rather refined) slightly by setting  $\ell = \frac{N}{2} \left(1 - \frac{d^{\frac{\delta}{\tau}} - 1}{d^{\frac{\delta}{\tau} + 1}}\right)$  so that  $\ell$  is now well-defined for any  $\tau \geq 1$ . The analysis of [11] works fine with this choice of  $\ell$ . Also, note that for larger values of  $\tau$ , the quantities  $\frac{d^{\frac{\delta}{\tau}} - 1}{d^{\frac{\delta}{\tau} + 1}}$  and  $\frac{k \ln d}{d}$  are close to each other as,

$$\lim_{\tau \rightarrow \infty} \frac{(d^{\frac{\delta}{\tau}} - 1) \cdot \tau}{(d^{\frac{\delta}{\tau}} + 1) \cdot \delta \ln d} = \frac{1}{2}.$$

In the follow-up work of [15], the class of  $\tau$ -supported homogeneous  $\Sigma\Pi\Sigma\Pi$  circuits occurs implicitly. It follows from their work that the above lower bound is in fact valid for the family of iterated matrix multiplication polynomial which is in VP (in fact is complete for a subclass of VP called algebraic branching programs).

► **Theorem 4.4.** [15]. *There exists an explicit family  $\{f_N\}$  of homogeneous degree  $d$  polynomials on  $N = d^{O(1)}$  variables in VP such that any  $\tau$ -supported homogeneous  $\Sigma\Pi\Sigma\Pi$  circuit computing  $f_N$  has top fanin at least  $N^{\Omega(\frac{d}{\tau})}$ .*

Combining Proposition 4.2 with the above theorem immediately yields theorem 1.1. In the next section we move on investigating homogeneous  $\Sigma\Pi\Sigma\Pi\Sigma^{[\tau]}$  circuits.

## 5 The lower bound for homogeneous $\Sigma\Pi\Sigma\Pi\Sigma^{[N^\mu]}$ formulas

Here we follow the outline given in section 2 and derive a lower bound for homogeneous  $\Sigma\Pi\Sigma\Pi\Sigma^{[N^\mu]}$ -formulas.

**Step 1: an upper bound for homogeneous  $\Sigma\Pi\Sigma\Pi\Sigma^{[N^\mu]}$ -formulas.** Let  $0 \leq \mu < 1$  be a fixed constant. Consider a homogeneous  $\Sigma\Pi\Sigma\Pi\Sigma^{[N^\mu]}$  formula of size  $s$  as in equation (2) computing a homogeneous  $N$ -variate polynomial of degree  $d$ . We pick a random set  $R \subseteq [N]$  by picking each variable independently at random with probability  $1 - p$ , where  $p = d^{-\beta}$  (for a suitable constant  $\beta > 0$ ), and upper bound the DPSP-complexity of  $\sigma_R(D(\mathbf{x}))$ .

► **Lemma 5.1.** *Let  $t = \sqrt{d}$ ,  $\alpha = \frac{2\mu+1}{1-\mu}$  and  $d^{2+\alpha} \leq N \leq 2d^{2+\alpha}$  be an integer. If  $s \leq N^{\frac{0.03}{2+\alpha}} \cdot \sqrt{d}$  then there exists a constant  $0 < \beta < \alpha$  such that with probability at least  $1 - \frac{1}{N^{\Omega(\sqrt{d})}}$ , a random restriction  $\sigma_R$  satisfies:*

$$\text{DPSP}_{k,\ell}(\sigma_R(D(\mathbf{x}))) \leq s \cdot \binom{\frac{d}{t} + 1}{k} \cdot \binom{N}{\ell + 2kt} \quad \text{for all } k, \ell \geq 0 \quad \text{satisfying } \ell + 2kt \leq \frac{N}{2}. \quad (8)$$

We defer the proof of this lemma to section 6.

**Step 2.1: constructing a suitable family of polynomials.** The explicit family of polynomials for which we prove the lower bound is a variant of the Nisan-Wigderson design based polynomials used in [13, 11, 15]. The choice of this family depends on the bottom fanin of the depth 5 formulas. When the bottom fanin is  $\tau = N^\mu$ , for some fixed  $0 \leq \mu < 1$ , the family is defined as follows. For an integer  $d$  and  $\alpha = \frac{2\mu+1}{1-\mu}$ , let  $q$  be the smallest prime number between  $d^{1+\alpha}$  and  $2d^{1+\alpha}$  (such a prime is guaranteed to exist by the Bertrand-Chebyshev theorem [3])<sup>18</sup>. We define a family of Nisan-Wigderson polynomials of degree  $d$  on  $N = d \cdot q$  variables, parametrized by a number  $r$  (to be fixed later in the analysis).

$$\text{NW}_r(x_{1,1}, x_{1,2}, \dots, x_{d,q}) := \sum_{\substack{h(z) \in \mathbb{F}_q[z] \\ \deg(h) \leq r}} \prod_{i \in [d]} x_{i, h(i)},$$

where  $\mathbb{F}_q$  is the finite field with  $q$  elements.

**Step 2.2: lower bounding the DPSP-complexity of our polynomial family.** For appropriate choices of integers  $r, k, \ell$  and a random restriction  $\sigma_R$ , we show that  $\text{DPSP}_{k,\ell}(\sigma_R(\text{NW}_r))$  is large with high probability.

► **Lemma 5.2 (The main technical lemma.).** *Let  $\text{NW}_r$  be the Nisan-Wigderson design based polynomial defined above. Suppose  $R$  is a set formed by picking each variable independently at random with probability  $1 - p$ , where  $p = d^{-\beta}$  and  $\beta > 0$  is any constant less than  $\alpha$ . Over any field  $\mathbb{F}$  of characteristic zero, for  $r = \frac{\alpha+\beta}{2(1+\alpha)} \cdot d - 1$ ,  $k = \delta \cdot \sqrt{d}$  (for a small constant  $\delta > 0$ ) and  $\ell = \frac{N}{2} (1 - \frac{k \ln d}{d})$ , we have*

$$\text{DPSP}_{k,\ell}(\sigma_R(\text{NW}_r)) \geq \frac{1}{d^{O(1)}} \min \left( \frac{p^k}{4^k} \cdot \binom{N}{k} \cdot \binom{N}{\ell}, \binom{N}{\ell + d - k} \right), \quad (9)$$

with probability at least  $1 - \frac{1}{d^{\Theta(1)}}$ .

We will prove this lemma in Section A of the appendix.

<sup>18</sup> We are avoiding ceil/floor notations for simplicity of exposition

**Final Step: comparing the two bounds.** Comparing the probabilities with which equations (8) and (9) are satisfied, we see that there exists a set  $R$  such that both of them are simultaneously satisfied, implying:

$$\begin{aligned} s &\geq \frac{\text{DPSP}_{k,\ell}(\sigma_R(\text{NW}_r))}{\binom{\frac{d}{t}+1}{k} \cdot \binom{N}{\ell+2kt}} \\ &= N^{\Omega(\sqrt{d})} \quad (\text{for small enough constant } \delta) \end{aligned}$$

The above implication can be worked out using the numerical estimates given in lemma 3.1. This proves the lower bound of theorem 1.2.

## 6 Upper bounding the measure for homogeneous $\Sigma\Pi\Sigma\Pi\Sigma^{[\tau]}$ formulas

Let  $D(\mathbf{x})$  be a homogeneous  $\Sigma\Pi\Sigma\Pi\Sigma^{[\tau]}$  formula with bottom fanin bounded by  $\tau = N^\mu$  where  $\mu \in [0, 1)$  is a fixed constant.

$$D(\mathbf{x}) = \sum_i \prod_j \sum_r Q_{ijr}, \tag{10}$$

where  $Q_{ijr}$  is a product of linear forms. As before, let  $\alpha = \frac{2\mu+1}{1-\mu}$ . In this section we give a proof of lemma 5.1. We first show that when we apply a random restriction to a small homogeneous  $\Sigma\Pi\Sigma\Pi\Sigma^{[N^\mu]}$  formula, then with high probability it decomposes into two pieces which are individually much easier to deal with.

► **Lemma 6.1** (Decomposition under random restrictions). *Suppose that  $D(\mathbf{x})$  has size  $s \leq N^{\frac{0.03}{2+\alpha} \cdot \sqrt{d}}$ . Then, it is possible to fix a constant  $0 < \beta < \alpha$  and<sup>19</sup> form a set  $R$  by picking each variable independently at random with probability  $1 - p$ , where  $p = d^{-\beta}$ , such that with probability at least  $1 - \frac{1}{N^{\Omega(\sqrt{d})}}$  the following is true:*

$$\sigma_R(D(\mathbf{x})) = D_1(\mathbf{x}) + D_2(\mathbf{x}),$$

where  $D_1(\mathbf{x})$  is a homogeneous  $\Sigma\Pi\Sigma\Pi^{[2\sqrt{d}]}$  formula having top fanin same as that of  $D(\mathbf{x})$ , and  $\text{DPSP}_{k,\ell}(D_2(\mathbf{x})) = 0$  for any choice of  $k$  and  $\ell$ .

Before proving this, let us see why it implies the required upper bound of lemma 5.1.

**Proof of Lemma 5.1.** Using the decomposition lemma 6.1, with probability at least  $1 - \frac{1}{N^{\Omega(\sqrt{d})}}$  we have:

$$\text{DPSP}_{k,\ell}(\sigma_R(D(\mathbf{x}))) \leq \text{DPSP}_{k,\ell}(D_1(\mathbf{x})).$$

Let  $t = \sqrt{d}$  and  $k, \ell$  be arbitrary integers satisfying  $\ell + 2kt \leq \frac{N}{2}$ . Then the dimension of the projected shifted partials of  $D_1(\mathbf{x})$  is upper bounded as in [11],

$$\text{DPSP}_{k,\ell}(\sigma_R(D(\mathbf{x}))) \leq s \cdot \binom{\frac{d}{t}+1}{k} \cdot \binom{N}{\ell+2kt}. \tag{11}$$

This proves lemma 5.1. ◀

### 6.1 Proof of the decomposition lemma

We will prove lemma 6.1 here by considering two cases separately:  $0 \leq \mu \leq \frac{1}{5}$  and  $\frac{1}{5} < \mu < 1$ . Let  $t = \sqrt{d}$ .

<sup>19</sup>The requirement of  $\beta < \alpha$  in the statement of lemma 6.1 comes from Lemma 5.2.

**Case 1.** Suppose  $0 \leq \mu \leq \frac{1}{5}$ . In this case the analysis is similar to the one outlined in Section 2. Let  $Q_{ijr}$  be a product of linear forms as in equation (10) and  $\deg(Q_{ijr}) > 2t$ . Then  $Q_{ijr}$  can be expressed as  $Q_{ijr} = \tilde{Q}_{ijr} \cdot P_{ijr}$  such that  $\deg(\tilde{Q}_{ijr}) = 2t$ , by simply multiplying out  $2t$  linear forms in  $Q_{ijr}$ . Since the support of every linear form in  $Q_{ijr}$  is bounded by  $\tau = N^\mu$ , the number of monomials in  $\tilde{Q}_{ijr}$  is bounded by  $\tau^{2t} = (N^\mu)^{2t}$ . The monomials of  $\tilde{Q}_{ijr}$  are of two types – those with individual degree of every variable bounded by 2 (and hence has support at least  $t$ ), and those with at least one variable of degree 3 or more.

Let  $R$  be a set formed by picking every variable independently at random with probability  $1 - p$ , where  $p = d^{-\beta}$  for an appropriate choice of  $\beta$  (to be fixed shortly). The probability that any monomial of support at least  $t$  in  $\tilde{Q}_{ijr}$  survives under the random restriction  $\sigma_R$  is bounded by  $p^t \cdot (N^\mu)^{2t}$ . Running over all  $Q_{ijr}$  in equation (10), with probability at least  $1 - s \cdot p^t \cdot (N^\mu)^{2t}$ ,

$$\sigma_R(D(\mathbf{x})) = \sum_i \prod_j \sum_{\substack{r \\ \deg(Q_{ijr}) \leq 2t}} \sigma_R(Q_{ijr}) + P,$$

where every monomial in  $P$  has a variable of degree 3 or more. Naturally,  $\text{DPSP}_{k,\ell}(P) = 0$  for any choice of  $k$  and  $\ell$ . Since  $s \leq N^{\frac{0.03}{2+\alpha} \cdot \sqrt{d}}$ ,  $p = d^{-\beta}$ ,  $\alpha = \frac{2\mu+1}{1-\mu}$  and  $t = \sqrt{d}$ , the “bad” probability is

$$\begin{aligned} s \cdot p^t \cdot (N^\mu)^{2t} &\leq (N^{\frac{0.03}{2+\alpha}} \cdot d^{-\beta} \cdot N^{2\mu})^t \\ &\leq (N^{\frac{0.03}{2+\alpha}} \cdot N^{-\frac{\beta}{2+\alpha}} \cdot 2^{\frac{\beta}{2+\alpha}} \cdot N^{2\mu})^t, \quad \text{as } \left(\frac{N}{2}\right)^{\frac{1}{2+\alpha}} \leq d \leq N^{\frac{1}{2+\alpha}} \end{aligned}$$

The above quantity is at most  $\frac{1}{N^{\Omega(\sqrt{d})}}$  if

1.  $2\mu + \frac{0.03}{2+\alpha} < \frac{\beta}{2+\alpha}$ , and
2.  $0 < \beta < \alpha$ .

It is easy to verify that these two conditions are satisfied if  $\beta = \frac{6.5\mu+0.03}{1-\mu}$  and considering  $\mu \leq \frac{1}{5}$ .

**Case 2.** Suppose  $\frac{1}{5} < \mu < 1$ . In this case we apply the random restriction in two phases.

**Phase 1:** Pick each variable independently at random with probability  $1 - p_1$ , where  $p_1 = d^{-\beta_1}$ , and form a set  $R_1$ . ( $\beta_1$  will be fixed shortly.) Let  $g$  be a linear form in a product  $Q_{ijr}$ . Assume without loss of generality that the support of  $g$  is exactly  $\tau = N^\mu$  (if not, simply fill in  $g$  with variables having zero coefficients). Then, the expected value of the support size of  $\sigma_{R_1}(g)$  is

$$\gamma := \mathcal{E}[\text{support size of } g] = d^{-\beta_1} \cdot N^\mu.$$

By Chernoff bound,

$$\Pr\{\text{bottom fanin of } \sigma_{R_1}(D(\mathbf{x})) \geq (1 + \sqrt{3}) \cdot \gamma\} \leq s \cdot e^{-\gamma}.$$

One can verify that the above probability is less than  $\frac{1}{N^{\Omega(\sqrt{d})}}$  if

$$\mu \cdot (2 + \alpha) > \beta_1 + \frac{1}{2}, \tag{12}$$

as  $s \leq N^{\frac{0.03}{2+\alpha} \cdot \sqrt{d}}$ . We will set  $\beta_1$  shortly to satisfy the above condition.

**Phase 2:** Pick each variable independently at random (and independent of Phase 1) with probability  $1 - p_2$ , where  $p_2 = d^{-\beta_2}$ , and form a set  $R_2$ . ( $\beta_2$  will be set to an appropriate value shortly.) We wish to study the formula  $\sigma_{R_2}(\sigma_{R_1}(D(\mathbf{x}))) = \sigma_{R_1 \cup R_2}(D(\mathbf{x}))$ .

If we set  $\beta_1$  satisfying equation (12) then with high probability the bottom fanin of  $\sigma_{R_1}(D(\mathbf{x}))$  is less than  $(1 + \sqrt{3}) \cdot \gamma$  – assume that this happens after Phase 1. The argument from here on is similar to that in Case 1. Let

$$\sigma_{R_1}(D(\mathbf{x})) = \sum_i \prod_j \sum_r Q'_{ijr},$$

where each linear form in every  $Q'_{ijr}$  has support size bounded by  $(1 + \sqrt{3}) \cdot \gamma$ . If  $\deg(Q'_{ijr}) \geq 2t$  then  $Q'_{ijr} = \tilde{Q}'_{ijr} \cdot P'_{ijr}$  where  $\deg(\tilde{Q}'_{ijr}) = 2t$  and number of monomial in  $\tilde{Q}'_{ijr}$  is bounded by  $(1 + \sqrt{3})^{2t} \cdot \gamma^{2t}$ . Once again, focus on those monomials in  $\tilde{Q}'_{ijr}$  that have support at least  $t$ . (Each of the remaining monomials in  $\tilde{Q}'_{ijr}$  has a variable of degree 3 or more.) The probability that any of those monomials in  $\tilde{Q}'_{ijr}$  survives after the random restriction  $\sigma_{R_2}$  is applied is bounded by  $p_2^t \cdot (1 + \sqrt{3})^{2t} \cdot \gamma^{2t}$ . Hence with probability at least  $1 - s \cdot p_2^t \cdot (1 + \sqrt{3})^{2t} \cdot \gamma^{2t}$ ,

$$\sigma_{R_1 \cup R_2}(D(\mathbf{x})) = \sigma_{R_2}(\sigma_{R_1}(D(\mathbf{x}))) = \sum_i \prod_j \sum_{\deg(Q'_{ijr}) \leq 2t} \sigma_{R_2}(Q'_{ijr}) + P',$$

where  $\text{DPSP}_{k,\ell}(P') = 0$  for any  $k, \ell$ . Let us calculate the bad probability a bit more closely.

$$\begin{aligned} s \cdot p_2^t \cdot (1 + \sqrt{3})^{2t} \cdot \gamma^{2t} &\leq [N^{\frac{0.03}{2+\alpha}} \cdot p_2 \cdot (1 + \sqrt{3})^2 \cdot \gamma^2]^t \\ &= [N^{\frac{0.03}{2+\alpha}} \cdot d^{-\beta_2} \cdot (1 + \sqrt{3})^2 \cdot d^{-2\beta_1} \cdot N^{2\mu}]^t. \end{aligned}$$

The above quantity is less than  $\frac{1}{N^{\Omega(\sqrt{d})}}$  if

$$2\mu \cdot (2 + \alpha) + 0.03 < \beta_2 + 2\beta_1, \quad \text{and} \tag{13}$$

$$\beta_1 + \beta_2 < \alpha \quad \& \quad \beta_1, \beta_2 > 0 \tag{14}$$

The requirement stated in equation (14) comes from Lemma 5.2, as Phase 1 and 2 together amounts to setting each variable zero independently with probability  $1 - p_1 p_2 = 1 - d^{-(\beta_1 + \beta_2)}$ . It is easy to verify that the conditions stated by equations (12), (13) and (14) are satisfied by choosing

$$\beta_1 = \mu \cdot (2 + \alpha) - 0.51$$

$$\beta_2 = 1.06,$$

and keeping in mind that  $\mu > \frac{1}{5}$ .

This completes the proof of the decomposition lemma.

## 7 Summary and discussion

A recent line of research on arithmetic circuit lower bounds uses the dimension of the space of shifted partials and its variant the projected shifted partials under random restriction as a complexity measure to make progress on proving lower bounds for certain interesting classes of arithmetic circuits, namely regular formulas and homogeneous depth four formulas. (The dimension of the space of shifted partials measure is in turn based on the classical measure of

the dimension of the space of partial derivatives.) The formal degree of a homogeneous depth four formula (or a regular formula) is bounded by the degree (or the order of the degree) of the polynomial that it computes. At this point it was not clear if the present techniques are applicable to models where the formal degree is much higher than the degree of the computed polynomial. One very interesting (and arguably the simplest nontrivial) example of such an unrestricted formal degree model is (nonhomogeneous) depth three circuits over fields of characteristic zero – its power being exhibited by the recent work of [7].

Our work takes a step forward in this direction by showing an exponential lower bound for (nonhomogeneous) depth three circuits with small bottom fanin over fields of characteristic zero. Along the way we also show an exponential lower bound for homogeneous depth five formulas with small bottom fanin. The second result is for an explicit polynomial in VNP. An immediate question is whether the combinatorial argument from [15] can be suitably adapted so that the lower bound of theorem 1.2 holds for iterated matrix multiplication as well. Both these results are obtained by building upon the current techniques on shifted partials based measures. It would be very interesting to prove analogous lower bounds for less restrictive subclasses of arithmetic circuits.

- Can we drop the restriction of ‘small bottom fanin’ from both the models – (nonhomogeneous) depth three circuits and homogeneous depth five circuits – and still show an exponential lower bound?

A few other intriguing problems on arithmetic circuit lower bounds are worth mentioning here:

- Show a super-polynomial lower bound for homogeneous bounded depth arithmetic circuits.
- Show a super-polynomial lower bound for homogeneous arithmetic formulas.
- Show a super-polynomial separation between homogeneous product-depth- $\Delta$  formulas and homogeneous product-depth- $(\Delta - 1)$  formulas.
- Solve the above problems without the assumption of homogeneity.

Solutions to these problems, using present or new techniques, would give a significant boost to our understanding of arithmetic circuit lower bounds.

**Acknowledgements.** The authors would like to thank Amit Chakrabarti, Mrinal Kumar, Satya Lokam and Ramprasad Saptharishi for helpful discussions. In particular, Ramprasad pointed out to us that a lemma in [7] can be improved quantitatively and that the  $\Sigma\Pi\Sigma$  circuits which come out of the depth reduction in [7] in fact have small bottom fanin.

---

## References

- 1 Manindra Agrawal and V. Vinay. Arithmetic circuits: A chasm at depth four. In *FOCS*, pages 67–75, 2008.
- 2 Noga Alon. Perturbed Identity Matrices Have High Rank: Proof and Applications. *Combinatorics, Probability & Computing*, 18(1-2):3–15, 2009.
- 3 Paul Erdős. Beweis eines Satzes von Tschebyschef. *Acta Sci. Math. (Szeged)*, 5:194–198, 1930-1932.
- 4 Hervé Fournier, Nutan Limaye, Guillaume Malod, and Srikanth Srinivasan. Lower bounds for depth 4 formulas computing iterated matrix multiplication. In *STOC*, pages 128–135, 2014.
- 5 Dima Grigoriev and Marek Karpinski. An exponential lower bound for depth 3 arithmetic circuits. In *STOC*, pages 577–582, 1998.

- 6 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.
- 7 Ankit Gupta, Pritish Kamath, Neeraj Kayal, and Ramprasad Saptharishi. Arithmetic circuits: A chasm at depth three. In *FOCS*, pages 578–587, 2013.
- 8 Ankit Gupta, Neeraj Kayal, Pritish Kamath, and Ramprasad Saptharishi. Approaching the chasm at depth four. In *CCC*, 2013.
- 9 G. H. Hardy and S. Ramanujan. Asymptotic formulae in combinatory analysis. *Proceedings of the London Mathematical Society*, s2-17(1):75–115, 1918.
- 10 Neeraj Kayal. An exponential lower bound for the sum of powers of bounded degree polynomials. Technical report, Electronic Colloquium on Computational Complexity (ECCC), 2012.
- 11 Neeraj Kayal, Nutan Limaye, Chandan Saha, and Srikanth Srinivasan. An exponential lower bound for homogeneous depth four arithmetic formulas. In *FOCS*, pages 61–70, 2014.
- 12 Neeraj Kayal, Nutan Limaye, Chandan Saha, and Srikanth Srinivasan. Super-polynomial lower bounds for depth-4 homogeneous arithmetic formulas. In *STOC*, pages 119–127, 2014.
- 13 Neeraj Kayal, Chandan Saha, and Ramprasad Saptharishi. A super-polynomial lower bound for regular arithmetic formulas. In *STOC*, pages 146–153, 2014.
- 14 Pascal Koiran. Arithmetic circuits: The chasm at depth four gets wider. *Theoretical Computer Science*, 448:56–65, 2012.
- 15 Mrinal Kumar and Shubhangi Saraf. On the power of homogeneous depth 4 arithmetic circuits. In *FOCS*, pages 364–373, 2014.
- 16 Mrinal Kumar and Shubhangi Saraf. Superpolynomial lower bounds for general homogeneous depth 4 arithmetic circuits. In *ICALP (1)*, pages 751–762, 2014.
- 17 D.E. Littlewood. *The Theory of Group Characters and Matrix Representations of Groups*. Ams Chelsea Publishing. AMS Chelsea Pub., 2nd edition, 1950.
- 18 Noam Nisan and Avi Wigderson. Lower bounds on arithmetic circuits via partial derivatives. *Computational Complexity*, 6(3):217–234, 1997. Available at <http://www.math.ias.edu/~avi/PUBLICATIONS/MYPAPERS/NW96/final.pdf>.
- 19 H. J. Ryser. Combinatorial mathematics. *Math. Assoc. of America*, 14, 1963.
- 20 Ramprasad Saptharishi. Personal communication, 2014.
- 21 Amir Shpilka. *Lower Bounds for Small Depth Arithmetic and Boolean Circuits*. PhD thesis, The Hebrew University, 2001.
- 22 Amir Shpilka and Avi Wigderson. Depth-3 arithmetic formulae over fields of characteristic zero. In *IEEE Conference on Computational Complexity*, pages 87–, 1999. Available at <http://eccc.hpi-web.de/report/1999/023/>.
- 23 Amir Shpilka and Avi Wigderson. Depth-3 arithmetic circuits over fields of characteristic zero. *Computational Complexity*, 10(1):1–27, 2001.
- 24 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.
- 25 Sébastien Tavenas. Improved bounds for reduction to depth 4 and depth 3. In *MFCS*, pages 813–824, 2013.

## **A** Proof of Lemma 5.2

In this section we prove lemma 5.2, i.e. we show that the dimension of projected shifted partial derivatives of a randomly restricted Nisan-Wigderson design based polynomial is within a “small” factor of the maximum possible with high probability. Our proof is very



similar to the proof of Lemma 13 in [11] – in fact, we reuse quite a bit of the argument from there but carefully tune it at places to achieve the required setting of parameters. Proofs of some of the propositions in this section are collected in Section B. Let  $e \stackrel{\text{def}}{=} (d-k)$  throughout the rest of this section.

**Preliminaries.** Note that in the construction in Section 5 of  $NW_r$ , there is a 1-1 correspondence between the variable indices in  $[N]$  and points in  $[d] \times [q]$ . Being homogeneous and multilinear of degree  $d$ , the monomials of  $NW_r$  are in 1-1 correspondence with sets in  $\binom{[N]}{d} \equiv \binom{[d] \times [q]}{d}$ . Indeed, from the construction it is clear that the coefficient of any monomial in  $NW_r$  is either 0 or 1 and that there is a 1-1 correspondence between monomials in the support of  $NW_r$  and univariate polynomials of degree at most  $r$  in  $\mathbb{F}_q[z]$ . Now since two distinct polynomials of degree  $r$  over a field have at most  $r$  common roots we get:

► **Proposition 1.1** (A basic property of our construction). *For any two distinct sets  $D_1, D_2 \in \binom{[d] \times [q]}{d}$  in the support of  $NW_r$ , we have*

$$|D_1 \cap D_2| \leq r.$$

Let  $R$  be a set formed by picking each variable independently at random with probability  $1-p$ , where  $p = d^{-\beta}$  for  $0 < \beta < \alpha$ . Our goal for the remainder of this section is to lower bound  $\text{DPSP}_{k,\ell}(\sigma_R(NW_r))$ .

**Reformulating our goal in terms of the rank of an explicit matrix.** Let  $f$  be any homogeneous multilinear polynomial of degree  $d$  on  $N$  variables. Then we have

$$\partial_{\text{ML}}^{\leq k} f = \left\{ \partial^C f : C \in \binom{[N]}{k} \right\}.$$

Note that every  $k$ -th order derivative of  $f$  is homogeneous and multilinear of degree  $(d-k)$ . Hence

$$\pi(\mathbf{x}^{\leq \ell} \cdot \partial_{\text{ML}}^{\leq k} f) = \left\{ \mathbf{x}_A \cdot \sigma_A(\partial^C f) : A \in \binom{[N]}{\ell}, C \in \binom{[N]}{k} \right\}.$$

Thus we have

► **Proposition 1.2.** *For any homogeneous multilinear polynomial  $f$  of degree  $d$  on  $N$  variables and for all integers  $k$  and  $\ell$ :*

$$\text{DPSP}_{k,\ell}(f) = \dim \left( \left\{ \mathbf{x}_A \cdot \sigma_A(\partial^C f) : A \in \binom{[N]}{\ell}, C \in \binom{[N]}{k} \right\} \right).$$

Now the  $\mathbb{F}$ -linear dimension of any set of polynomials is the same as the rank of the matrix corresponding to our set of polynomials in the natural way. In fact, we will focus our attention on a subset of rows of this matrix and prove a lower bound on the rank of the matrix defined by this subset of rows. Specifically,

► **Proposition 1.3.** *Let  $f$  be a homogeneous multilinear polynomial of degree  $d$  on  $N$  variables. Let  $k, \ell$  be integers. Define a matrix  $M(f)$  as follows. The rows of  $M(f)$  are labelled by pairs of subsets  $(A, C) \in \binom{[N]}{\ell} \times \binom{[N]}{k}$  such that  $A \cap C = \Phi$  (null set) and columns are indexed by subsets  $S \in \binom{[N]}{\ell+e}$ . Each row  $(A, C)$  corresponds to the polynomial*

$$f_{A,C} \stackrel{\text{def}}{=} \mathbf{x}_A \cdot \sigma_A(\partial^C f)$$

*in the following way. The  $S$ -th entry of the row  $(A, C)$  is the coefficient of  $\mathbf{x}_S$  in the polynomial  $f_{A,C}$ . Then,*

$$\text{DPSP}_{k,\ell}(f) \geq \text{rank}(M(f)).$$



So our problem is equivalent to lower bounding the rank of the matrix  $M(f)$  for our constructed polynomial  $f$ . Now note that the entries of  $M(f)$  are coefficients of appropriate monomials of  $f$  and it will be helpful to us in what follows to keep track of this information. We will do it by assigning a label to each cell of  $M(f)$  as follows. We will think of every location in the matrix  $M(f)$  being labelled with either a set  $D \in \binom{[N]}{d}$  or the label `InvalidSet` depending on whether that entry contains the coefficient of the monomial  $\mathbf{x}_D$  of  $f$  or it would have been zero regardless of the actual coefficients of  $f$ . Specifically, let us introduce the following notation. For sets  $A, B$  define:

1.

$$A \parallel B = \begin{cases} A \setminus B & \text{if } B \subseteq A \\ \text{InvalidSet} & \text{otherwise} \end{cases}$$

2.

$$A \uplus B = \begin{cases} A \cup B & \text{if } B \cap A = \emptyset \\ \text{InvalidSet} & \text{otherwise} \end{cases}$$

Then the label of the  $((A, C), S)$ -th cell of  $M(f)$  is defined to be the set  $(S \parallel A) \uplus C$ . Equivalently, if the label of a cell of the  $(A, C)$ -th row of  $M$  is a set  $D$  then the column must be the one corresponding to  $S = (D \parallel C) \uplus A$  (if  $C$  is not a subset of  $D$  or if  $D$  and  $A$  are not disjoint then  $D$  cannot occur in the row indexed by  $(A, C)$ ). For the rest of this section, we will refer to  $M(\sigma_R(\text{NW}_r))$  simply as the matrix  $M$ . Our goal then is to show that the rank of this matrix  $M$  is reasonably close to the trivial upper bound, viz. the minimum of the number of rows and the number of columns of  $M$  with high probability. It turns out that our matrix  $M$  is a relatively sparse matrix and we will exploit this fact by using a relevant lemma from real matrix analysis to obtain a lower bound on its rank.

**The Surrogate Rank.** Consider the matrix  $B \stackrel{\text{def}}{=} M^T \cdot M$ . Then  $B$  is a real symmetric, positive semidefinite matrix. From the definition of  $B$  it is easy to show that:

► **Proposition 1.4.** *Over any field  $\mathbb{F}$  we have*

$$\text{rank}(B) \leq \text{rank}(M).$$

*Over the field  $\mathbb{R}$  of real numbers we have*

$$\text{rank}(B) = \text{rank}(M).$$

So it suffices to lower bound the rank of  $B$ . By an application of Cauchy-Schwarz on the vector of nonzero eigenvalues of  $B$ , one obtains:

► **Lemma 1.5** ([2]). *Over the field of real numbers  $\mathbb{R}$  we have:*

$$\text{rank}(B) \geq \frac{\text{Tr}(B)^2}{\text{Tr}(B^2)}.$$

Let us call the quantity  $\frac{\text{Tr}(B)^2}{\text{Tr}(B^2)}$  as the surrogate rank of  $B$ , denoted  $\text{SurRank}(B)$ . It then suffices to show that this quantity is within a ‘small’ factor of  $U = \min\left(\binom{N}{\ell+e}, \binom{N}{\ell} \cdot \binom{N}{k}\right)$  with high probability. In the rest of this section, we will first derive an exact expression for  $\text{SurRank}(B)$  and then show that it is close to  $U$  (again, with high probability). In the following discussion we would need an estimate of a quantity  $R_d(w, r)$  that denotes the number of univariate polynomials in  $\mathbb{F}_q[z]$  of degree at most  $r$  having exactly  $w$  distinct roots in  $[d]$ .

**An estimate for  $R_d(w, r)$ .** First note that any polynomial  $h(z) \in \mathbb{F}_q[z]$  of degree at most  $r$  that has  $w$  roots in  $[d]$  must be of the form

$$h(z) = (z - \alpha_1) \cdot (z - \alpha_2) \cdot \dots \cdot (z - \alpha_w) \cdot \hat{h}(z),$$

where each  $\alpha_i$  is in  $[d]$  and  $\hat{h}(z) \in \mathbb{F}_q[z]$  is of degree at most  $(r - w)$ . Thus we have

$$R_d(w, r) \leq q^{r-w+1} \cdot \binom{d}{w} \leq q^{r+1} \cdot \left(\frac{d}{q}\right)^w \cdot \frac{1}{w!} \quad (15)$$

### A.1 Deriving an exact expression for $\text{SurRank}(B)$ .

We will now calculate an exact expression for  $\text{SurRank}(B)$ , or equivalently an exact expression for  $\text{Tr}(B)$  and  $\text{Tr}(B^2)$ .

**Calculating  $\text{Tr}(B)$ .** Calculating  $\text{Tr}(B)$  is fairly straightforward. From the definition of the matrix  $B$  we have:

► **Proposition 1.6.** *For any  $0, \pm 1$  matrix  $M$  (i.e. a matrix all of whose entries are either 0, or +1 or -1) we have*

$$\text{Tr}(B) = \text{Tr}(M^T \cdot M) = \text{number of nonzero entries in } M.$$

Now we can calculate the number of nonzero entries in  $M$  by going over all sets  $D \in \binom{[N]}{d} \cap \text{Supp}(\sigma_R(\text{NW}_r))$ , calculating the number of cells of  $M$  labelled with  $D$  and adding these up. Clearly

$$\sigma_R(\text{NW}_r) = \sum_{D \in \text{Supp}(\text{NW}_r)} e_D \cdot \mathbf{x}_D,$$

where  $e_D$  is an indicator variable such that  $e_D = 1$  if  $\sigma_R(\mathbf{x}_D) \neq 0$ , and  $e_D = 0$  otherwise. Hereafter, we will refer to  $\sigma_R(\text{NW}_r)$  as  $g$  at some places, and the number of monomials in  $\sigma_R(\text{NW}_r)$  as  $\mu(g)$ .

$$\begin{aligned} \mu(g) &= \sum_{D \in \text{Supp}(\text{NW}_r)} e_D \\ \Rightarrow \mathcal{E}[\mu(g)] &= p^d \cdot q^{r+1} = \gamma \text{ (say)} \\ \Rightarrow \mathcal{E}[\text{Tr}(B)] &= \gamma \cdot \binom{d}{k} \cdot \binom{N-d}{\ell}. \end{aligned}$$

► **Proposition 1.7.**  $\Pr \left[ \text{Tr}(B) \leq \frac{1}{2} \cdot \gamma \cdot \binom{d}{k} \cdot \binom{N-d}{\ell} \right] \leq \frac{10}{pd^\alpha}$ . (Proof in Section B)

**Calculating  $\text{Tr}(B^2)$ .** From the definition of  $B = M^T \cdot M$  and expanding out the relevant summations we get:

► **Proposition 1.8.**  $\text{Tr}(B^2)$  equals

$$\sum_{(A_1, C_1), (A_2, C_2) \in \left( \binom{[N]}{\ell} \times \binom{[N]}{k} \right)^2} \sum_{S_1, S_2 \in \binom{[N]}{\ell+e}} M_{(A_1, C_1), S_1} \cdot M_{(A_1, C_1), S_2} \cdot M_{(A_2, C_2), S_1} \cdot M_{(A_2, C_2), S_2}.$$

We will use the following notation in doing this calculation. For a pair of row indices  $((A_1, C_1), (A_2, C_2)) \in \left( \binom{[N]}{\ell} \times \binom{[N]}{k} \right)^2$  and a pair of column indices  $S_1, S_2 \in \binom{[N]}{\ell+e}$ , the

box  $\mathbf{b}$  defined by them, denoted  $\mathbf{b} = 2 - \text{box}((A_1, C_1), (A_2, C_2), S_1, S_2)$  is the four-tuple of cells

$$(((A_1, C_1), S_1), ((A_1, C_1), S_2), ((A_2, C_2), S_1), ((A_2, C_2), S_2)).$$

Since all the entries of our matrix  $M$  are either 0 or 1 we have:

► **Proposition 1.9.**

$$\text{Tr}(B^2) = \text{Number of boxes } \mathbf{b} \text{ with all four entries nonzero.}$$

For a box  $\mathbf{b} = 2 - \text{box}((A_1, C_1), (A_2, C_2), S_1, S_2)$ , its tuple of labels, denoted  $\text{labels}(\mathbf{b})$  is the tuple of labels of the cells  $((A_1, C_1), S_1), ((A_1, C_1), S_2), ((A_2, C_2), S_1), ((A_2, C_2), S_2)$  in that order. In other words,

$$\text{labels}(\mathbf{b}) = ((S_1 \setminus A_1) \uplus C_1, (S_2 \setminus A_1) \uplus C_1, (S_1 \setminus A_2) \uplus C_2, (S_2 \setminus A_2) \uplus C_2).$$

We then have

► **Proposition 1.10.**  $\text{Tr}(B^2)$  equals the number of boxes

$$\mathbf{b} = 2 - \text{box}((A_1, C_1), (A_2, C_2), S_1, S_2)$$

such that all the four labels in  $\text{labels}(\mathbf{b})$  are valid sets in the support of our design polynomial  $\sigma_R(\text{NW}_r)$ .

So our problem boils down to counting the number of boxes in which all the four labels are valid sets in the support of our polynomial  $\sigma_R(\text{NW}_r)$ . Let us analyze the box

$$\mathbf{b} = 2 - \text{box}((A_1, C_1), (A_2, C_2), S_1, S_2)$$

a bit closely. Suppose  $\text{labels}(\mathbf{b}) = (D_1, D_2, D_3, D_4)$  as shown in the table below where  $D_1, D_2, D_3, D_4$  are valid sets in  $\binom{[N]}{d}$ .

	$S_1$	$S_2$
$(A_1, C_1)$	$D_1$	$D_2$
$(A_2, C_2)$	$D_3$	$D_4$

Define the following sets:

$$\begin{aligned} E_1 &:= A_1 \setminus (A_1 \cap A_2) & E_2 &:= A_2 \setminus (A_1 \cap A_2) \\ E_3 &:= C_1 & E_4 &:= C_2 \\ E_5 &:= D_1 \setminus (E_2 \uplus E_3) & E_6 &:= D_2 \setminus (E_2 \uplus E_3) \\ &= D_3 \setminus (E_1 \uplus E_4) & &= D_4 \setminus (E_1 \uplus E_4) \end{aligned}$$

Note that  $E_2 \uplus E_3$  must be a subset of both  $D_1$  and  $D_2$ , similarly  $E_1 \uplus E_4$  must be a subset of both  $D_3$  and  $D_4$ . Also,  $D_1 \setminus (E_2 \uplus E_3) = D_3 \setminus (E_1 \uplus E_4)$  as  $(D_1 \setminus C_1) \uplus A_1 = (D_3 \setminus C_2) \uplus A_2 = S_1$ . Similarly,  $D_2 \setminus (E_2 \uplus E_3) = D_4 \setminus (E_1 \uplus E_4)$ . Verify that  $D_1, D_2, D_3$  and  $D_4$  can be expressed as:

$$\begin{aligned} D_1 &= E_2 \uplus E_5 \uplus E_3 & D_2 &= E_2 \uplus E_6 \uplus E_3 \\ D_3 &= E_1 \uplus E_5 \uplus E_4 & D_4 &= E_1 \uplus E_6 \uplus E_4 \end{aligned} \tag{16}$$

From the above definitions, if  $|A_1 \cap A_2| = v$  then

$$\begin{aligned} |E_1| &= |E_2| = \ell - v \\ |E_3| &= |E_4| = k \\ |E_5| &= |E_6| = d - (\ell - v + k) \end{aligned} \tag{17}$$

► **Proposition 1.11.** *Unless  $D_1, D_2, D_3, D_4$  are all distinct sets,  $\text{labels}(\mathbf{b})$  contains at most two distinct sets. Furthermore, if  $D_1, D_2, D_3$  are distinct then  $\ell - v + k \leq r$  and  $d - (\ell - v + k) \leq r$ .*

**Proof.** We show that if  $D_1$  equals any of  $D_2, D_3$  or  $D_4$  then  $\text{labels}(\mathbf{b})$  has at most two distinct sets. The argument is similar for other cases. Suppose  $D_1 = D_2$  then by Equation 16  $E_5 = E_6$ , implying  $D_3 = D_4$ . If  $D_1 = D_3$  then again by Equation 16,  $E_2 \uplus E_3 = E_1 \uplus E_4$  implying  $D_2 = D_4$ . Now suppose  $D_1 = D_4$ , then by Equation 16,  $E_6 \subseteq D_1$ . But  $E_6 \subseteq D_2$ , which means  $D_2 \subseteq D_1$  as  $E_2 \uplus E_3 \subseteq D_1$ . Since  $|D_2| = |D_1| = d$ ,  $D_1 = D_2$  and hence  $D_1 = D_2 = D_3 = D_4$ .

To prove the second statement of the lemma, observe that  $|D_1 \cap D_2| \geq |E_2 \uplus E_3| = \ell - v + k$ . So, if  $\ell - v + k \geq r + 1$  then  $D_1 = D_2$ . Similarly,  $|D_1 \cap D_3| \geq |E_5| = d - (\ell - v + k)$ . If  $d - (\ell - v + k) \geq r + 1$  then  $D_1 = D_3$ . ◀

This means that any box  $\mathbf{b}$  that contributes to  $\text{Tr}(B^2)$  must have the property that its label set  $\text{labels}(\mathbf{b})$  contains at most two distinct sets in the support of  $\sigma_R(\text{NW}_r)$ , or four distinct sets in the support of  $\sigma_R(\text{NW}_r)$ . A set  $D$  is in the support of  $\sigma_R(\text{NW}_r)$  if  $D$  is in the support of  $\text{NW}_r$  and  $\sigma_R(\mathbf{x}_D) \neq 0$ . (Recall that  $e_D$  is an indicator variable which is 1 if  $\sigma_R(\mathbf{x}_D) \neq 0$ , and zero otherwise.)

► **Corollary 1.12.** *For any four distinct sets  $D_1, D_2, D_3, D_4 \in \binom{[N]}{d}$  define*

$$\begin{aligned} \mu_0(D_1) &\stackrel{\text{def}}{=} \{\text{box } \mathbf{b} : \text{labels}(\mathbf{b}) = (D_1, D_1, D_1, D_1)\} \\ \mu_1(D_1, D_2) &\stackrel{\text{def}}{=} \{\text{box } \mathbf{b} : \text{labels}(\mathbf{b}) = (D_1, D_2, D_1, D_2)\} \\ \mu_2(D_1, D_2) &\stackrel{\text{def}}{=} \{\text{box } \mathbf{b} : \text{labels}(\mathbf{b}) = (D_1, D_1, D_2, D_2)\} \\ \mu_3(D_1, D_2, D_3, D_4) &\stackrel{\text{def}}{=} \{\text{box } \mathbf{b} : \text{labels}(\mathbf{b}) = (D_1, D_2, D_3, D_4)\} \end{aligned}$$

*Let the support of  $\text{NW}_r$ , denoted  $\text{Supp}(\text{NW}_r) \subset \binom{[N]}{d}$ , be the set of all sets  $D \in \binom{[N]}{d}$  such that the coefficient of the monomial  $\mathbf{x}_D$  in  $\text{NW}_r$  is nonzero. Define  $T_0, T_1, T_2, T_3$  as follows:*

$$\begin{aligned} T_0 &= \sum_{D_1 \in \text{Supp}(\text{NW}_r)} e_{D_1} \cdot |\mu_0(D_1)| \\ T_1 &= \sum_{D_1 \neq D_2 \in \text{Supp}(\text{NW}_r)} e_{D_1} \cdot e_{D_2} \cdot |\mu_1(D_1, D_2)| \\ T_2 &= \sum_{D_1 \neq D_2 \in \text{Supp}(\text{NW}_r)} e_{D_1} \cdot e_{D_2} \cdot |\mu_2(D_1, D_2)| \\ T_3 &= \sum_{D_1 \neq D_2 \neq D_3 \neq D_4 \in \text{Supp}(\text{NW}_r)} e_{D_1} \cdot e_{D_2} \cdot e_{D_3} \cdot e_{D_4} \cdot |\mu_3(D_1, D_2, D_3, D_4)| \end{aligned} \quad (18)$$

Then

$$\text{Tr}(B^2) = T_0 + T_1 + T_2 + T_3.$$

We are using the notation  $D_1 \neq D_2 \neq D_3 \neq D_4$  to mean that the four sets are distinct. The proof of Proposition 1.11 rules out the existence of any box  $\mathbf{b}$  having  $\text{labels}(\mathbf{b}) = (D_1, D_2, D_2, D_1)$  with distinct  $D_1, D_2 \in \text{Supp}(\text{NW}_r)$  and that is why there is no term in  $\text{Tr}(B^2)$  corresponding to such boxes.

Proposition 1.7 shows that  $\text{Tr}(B)$  is large with high probability. In order to lower bound  $\frac{\text{Tr}(B)^2}{\text{Tr}(B^2)}$ , we will show that  $\text{Tr}(B^2)$  is less than an upper bound with high probability. This is achieved by upper bounding the expected values of  $T_0, T_1, T_2$  and  $T_3$  and then applying Markov's inequality.

### A.2 Upper bound for $\mathcal{E}[T_3]$

Let  $\rho(D_1, D_2, D_3)$  be the number of pairs of rows  $((A_1, C_1), (A_2, C_2))$  in which  $D_1, D_2, D_3$  (all distinct) can possibly occur as labels (as depicted in the table before). For a fixed  $D_1, D_2, D_3$  we upper bound  $\rho(D_1, D_2, D_3)$  with the help of Equation 16. Notice that for a fixed  $D_1, D_2, D_3$ , if we specify  $E_2, E_3, E_4$  and  $A_1 \cap A_2$  then the sets  $A_1, C_1, A_2, C_2$  are determined. Let us count the number of ways we can pick  $E_2, E_3, E_4$  and  $A_1 \cap A_2$  for a given  $D_1, D_2, D_3$ . Taking the size bounds on the sets into account from Equation 17, this quantity is upper bounded by,

$$\binom{d}{\ell - v} \cdot \binom{d - (\ell - v)}{k} \cdot \binom{\ell - v + k}{k} \cdot \binom{N - d}{v}.$$

The quantity  $\binom{N-d}{v}$  is an upper bound on the number of ways we can pick  $A_1 \cap A_2$  as  $A_1$  must be disjoint from  $D_1$ . By Proposition 1.11,  $\ell - v + k \leq r < d$ , (also,  $v \leq \ell < \frac{N-d}{2}$ ) implying

$$\rho(D_1, D_2, D_3) \leq 2^d \cdot \binom{d}{k}^2 \cdot \binom{N - d}{\ell} = \rho \quad (\text{say}). \tag{19}$$

Hence,

$$T_3 \leq \rho \cdot \sum_{D_1 \neq D_2 \neq D_3 \in \text{Supp}(\text{NW}_r)} e_{D_1} \cdot e_{D_2} \cdot e_{D_3} \tag{20}$$

Now we upper bound the expected value of the quantity  $\sum_{D_1 \neq D_2 \neq D_3 \in \text{Supp}(\text{NW}_r)} e_{D_1} \cdot e_{D_2} \cdot e_{D_3} = \eta$  (say) in the following proposition.

► **Proposition 1.13.**  $\mathcal{E}[\eta] \leq 4 \cdot \gamma^2 \cdot q^{(r+1)} \cdot \left(\frac{d}{q}\right)^d$ , where  $\gamma$  is as in Proposition 1.7. This implies

$$\mathcal{E}[T_3] \leq 4 \cdot \left(\frac{2}{d^{\frac{\alpha-\beta}{2}}}\right)^d \cdot \gamma^2 \cdot \binom{d}{k}^2 \cdot \binom{N - d}{\ell}.$$

Proof of the above proposition is omitted. We show in the later sections that  $\mathcal{E}[T_3]$  is negligible compared to  $\mathcal{E}[T_0 + T_1 + T_2]$  and hence does not contribute much to the expected value of  $\text{Tr}(B^2)$ .

In what follows we will derive expressions for  $|\mu_0(D_1)|, |\mu_1(D_1, D_2)|$  and  $|\mu_2(D_1, D_2)|$  and compute expected values of  $T_0, T_1$  and  $T_2$  by summing these up over  $D_1, D_2 \in \text{Supp}(\sigma_R(\text{NW}_r))$ . We first observe:

► **Proposition 1.14.** For any set  $D_1 \in \binom{[N]}{d}$  and any row  $(A, C)$  of  $M$ , there can be at most one cell in that row labelled with the set  $D_1$ .

This means that any box  $\mathbf{b} = 2 - \text{box}((A_1, C_1), (A_2, C_2), S_1, S_2)$  contributing to either  $\mu_0(D_1)$  or  $\mu_2(D_1, D_2)$ , the columns  $S_1$  and  $S_2$  must be the same.

### A.3 Calculating $\mu_0(D_1)$ and $\mathcal{E}[T_0]$ .

Every box  $\mathbf{b} \in \mu_0(D_1)$  is of the form  $\mathbf{b} = 2 - \text{box}((A_1, C_1), (A_2, C_2), S_1, S_1)$  where both the entries  $((A_1, C_1), S_1)$  and  $((A_2, C_2), S_1)$  are both labelled by  $D_1$ . This implies  $A_1 = A_2$  and  $C_1 = C_2$ : By Equation 16,  $E_1 \subseteq D_3 = D_1$ , but  $A_1$  is disjoint from  $D_1$  and  $E_1 \subseteq A_1$ . Hence,  $E_1$  is an empty set and similarly  $E_2$  is also an empty set. This also implies  $E_3 = E_4$  from Equation 16 as  $D_3 = D_1$ . Analyzing this situation gives

► **Proposition 1.15.**

$$|\mu_0(D_1)| = \binom{N-d}{\ell} \cdot \binom{d}{k} \quad \text{and} \quad \mathcal{E}[T_0] = \gamma \cdot \binom{N-d}{\ell} \cdot \binom{d}{k}$$

**Proof.** For a fixed  $D_1$ , we can choose  $C_1$  in  $\binom{d}{k}$  ways and  $A_1$  in  $\binom{N-d}{\ell}$  ways. (Recall  $A_1$  must be disjoint from  $D_1$ .) The expression for  $\mathcal{E}[T_0]$  follows immediately from Equation 18. ◀

#### A.4 Calculating $\mu_1(D_1, D_2)$ and $\mathcal{E}[T_1]$ .

Let  $D_1, D_2 \in \binom{[N]}{d}$  be two distinct subsets in the support of  $NW_r$ . We consider a box  $\mathbf{b} = 2 - \text{box}((A_1, C_1), (A_2, C_2), S_1, S_2)$  in  $\mu_1(D_1, D_2)$ . Observe that even in this case it must be that  $A_1 = A_2$  and  $C_1 = C_2$ : By the same reason as before since  $D_3$  equals  $D_1$  in Equation 16. Analyzing this situation gives

► **Proposition 1.16.** *If  $|D_1 \cap D_2| = w$  then*

$$|\mu_1(D_1, D_2)| = \binom{N-2d+w}{\ell} \cdot \binom{w}{k} \quad \text{and hence} \quad \mathcal{E}[T_1] \leq d \cdot \frac{\gamma^2}{d^{(\alpha-\beta)k} \cdot k!} \cdot \binom{N-2d+k}{\ell}.$$

Proof of the above proposition is given in Section B.

#### A.5 Calculating $\mu_2(D_1, D_2)$ and $\mathcal{E}[T_2]$ .

Let  $D_1, D_2 \in \binom{[N]}{d}$  be two distinct subsets in the support of  $NW_r$ . We consider a box  $\mathbf{b} = 2 - \text{box}((A_1, C_1), (A_2, C_2), S_1, S_2)$  in  $\mu_2(D_1, D_2)$ . As we observed before this can happen only if  $S_1 = S_2 = S$  (say). Let  $|C_1 \cap C_2| = u$ . Analyzing this situation gives

► **Proposition 1.17.** *If  $|D_1 \cap D_2| = w$  then*

$$\begin{aligned} |\mu_2(D_1, D_2)| &= \sum_{0 \leq u \leq k} \binom{N-2d+w}{\ell-d+k+w-u} \cdot \binom{d-w}{k-u} \cdot \binom{d-w}{k-u} \cdot \binom{w}{u}, \text{ and hence} \\ \mathcal{E}[T_2] &\leq dk \cdot \gamma^2 \cdot \binom{N-2d}{\ell-d+k} \cdot \binom{d}{k}^2. \end{aligned}$$

**Proof.** The expectation calculation is similar to the one in the proof of Proposition 1.16 – the maxima of the relevant expression is touched at  $w = u = 0$ . ◀

#### A.6 Lower bound on $\text{SurRank}(B)$

A comparison between the binomial coefficients  $\binom{N-2d}{\ell-d+k}$  and  $\binom{N-d}{\ell}$  shows that

$$\binom{N-2d}{\ell-d+k} \geq \frac{1}{3^d} \cdot \binom{N-d}{\ell}.$$

Thus, from Proposition 1.15, 1.17 and 1.13, the upper bound on  $\mathcal{E}[T_2]$  dominates the upper bounds on  $\mathcal{E}[T_0]$  and  $\mathcal{E}[T_3]$ . Applying Markov's inequality,

$$\text{Tr}(B^2) \leq d^2 \cdot \frac{\gamma^2}{d^{(\alpha-\beta)k} \cdot k!} \cdot \binom{N-2d+k}{\ell} + 3d^2 k \cdot \gamma^2 \cdot \binom{N-2d}{\ell-d+k} \cdot \binom{d}{k}^2$$

with probability at least  $1 - \frac{1}{d}$ . Coupled with Proposition 1.7,

$$\text{SurRank}(B) \geq \min \left( \frac{\frac{1}{4} \cdot \gamma^2 \cdot \binom{d}{k}^2 \cdot \binom{N-d}{\ell}^2}{2d^2 \cdot \frac{\gamma^2}{d^{(\alpha-\beta)k} \cdot k!} \cdot \binom{N-2d+k}{\ell}}, \frac{\frac{1}{4} \cdot \gamma^2 \cdot \binom{d}{k}^2 \cdot \binom{N-d}{\ell}^2}{6d^2 k \cdot \gamma^2 \cdot \binom{N-2d}{\ell-d+k} \cdot \binom{d}{k}^2} \right),$$

with probability at least  $1 - \frac{1}{d^{\Omega(1)}}$ . The first ratio is at least  $\frac{p^k}{d^{O(1)}} \cdot \frac{1}{4^k} \cdot \binom{N}{k} \cdot \binom{N}{\ell}$  as

$$\frac{\binom{N-d}{\ell}^2}{\binom{N-2d+k}{\ell}} \geq \frac{1}{2^k d^{O(1)}} \cdot \binom{N}{\ell} \quad \text{and} \quad d^{\alpha k} \cdot k! \cdot \binom{d}{k}^2 \geq \frac{1}{2^k d^{O(1)}} \cdot \binom{N}{k}.$$

The second ratio is at least  $\frac{1}{d^{O(1)}} \cdot \binom{N}{\ell+d-k}$  as,

$$\frac{\binom{N-d}{\ell}^2}{\binom{N-2d}{\ell-d+k}} \geq \frac{1}{d^{O(1)}} \cdot \binom{N}{\ell+d-k}.$$

Therefore,

$$\text{SurRank}(B) \geq \frac{1}{d^{O(1)}} \min \left( \frac{p^k}{4^k} \cdot \binom{N}{k} \cdot \binom{N}{\ell}, \binom{N}{\ell+d-k} \right).$$

## B Proofs of certain propositions

**Proposition 1.7.**  $\Pr \left[ \text{Tr}(B) \leq \frac{1}{2} \cdot \gamma \cdot \binom{d}{k} \cdot \binom{N-d}{\ell} \right] \leq \frac{10}{pd^\alpha}.$

**Proof.** As in Proposition 1.6,  $\text{Tr}(B) = \text{Tr}(M^T \cdot M) =$  number of nonzero entries in  $M$ .

$$\begin{aligned} \text{Tr}(B) &= \mu(g) \cdot \binom{d}{k} \cdot \binom{N-d}{\ell} \\ \Rightarrow \mathcal{E}[\text{Tr}(B)] &= \gamma \cdot \binom{d}{k} \cdot \binom{N-d}{\ell} \end{aligned}$$

Hence,

$$\Pr \left[ \text{Tr}(B) \leq \frac{1}{2} \cdot \gamma \cdot \binom{d}{k} \cdot \binom{N-d}{\ell} \right] = \Pr \left[ \mu(g) \leq \frac{1}{2} \cdot \gamma \right].$$

It turns out that the variance of  $\mu(g)$ , denoted by  $\text{Var}(\mu(g))$ , can be upper bounded as follows.

$$\begin{aligned} \text{Var}(\mu(g)) &\leq \gamma \cdot (1 - p^d) + \gamma^2 \cdot \frac{2}{pd^\alpha} \quad (\text{proof omitted}) \\ \Rightarrow \Pr \left[ \mu(g) \leq \frac{1}{2} \cdot \gamma \right] &\leq \frac{10}{pd^\alpha} \quad (\text{by Chebyshev's inequality}) \end{aligned}$$

The last inequality also uses the fact that  $\gamma > 2pd^\alpha$  which is true since  $r = \frac{\alpha+\beta}{2(1+\alpha)} \cdot d - 1$  and hence  $\gamma = d^{\Omega(d)}$ . ◀

**Proposition 1.16.** If  $|D_1 \cap D_2| = w$  then

$$|\mu_1(D_1, D_2)| = \binom{N-2d+w}{\ell} \cdot \binom{w}{k} \quad \text{and hence} \quad \mathcal{E}[T_1] \leq d \cdot \frac{\gamma^2}{d^{(\alpha-\beta)k} \cdot k!} \cdot \binom{N-2d+k}{\ell}.$$

**Proof.** For a given  $D_1, D_2$ , let us count the number of rows  $(A, C)$  in which  $D_1$  and  $D_2$  can occur as labels. Since  $C \subset D_1 \cap D_2$  and  $|D_1 \cap D_2| = w$ , we can pick  $C$  in  $\binom{w}{k}$  ways. For

every choice of  $C$ , we can pick  $A$  in  $\binom{N-2d+w}{\ell}$  ways as  $A$  must be disjoint from  $D_1 \cup D_2$  and  $|D_1 \cup D_2| = 2d - w$ . By Equation 18,

$$\begin{aligned}
T_1 &= \sum_{D_1 \in \text{Supp}(\text{NW}_r)} \sum_{w \geq k} \sum_{\substack{D_2 \in \text{Supp}(\text{NW}_r) \\ D_2 \neq D_1, |D_2 \cap D_1| = w}} e_{D_1} \cdot e_{D_2} \cdot |\mu_1(D_1, D_2)| \\
\Rightarrow \mathcal{E}[T_1] &= \sum_{D_1 \in \text{Supp}(\text{NW}_r)} \sum_{w \geq k} \sum_{\substack{D_2 \in \text{Supp}(\text{NW}_r) \\ D_2 \neq D_1, |D_2 \cap D_1| = w}} p^d \cdot p^{d-w} \cdot \binom{N-2d+w}{\ell} \cdot \binom{w}{k} \\
&\leq p^{2d} \cdot \sum_{D_1 \in \text{Supp}(\text{NW}_r)} \sum_{w \geq k} R_d(w, r) \cdot p^{-w} \cdot \binom{N-2d+w}{\ell} \cdot \binom{w}{k} \\
&\leq p^{2d} \cdot \sum_{D_1 \in \text{Supp}(\text{NW}_r)} \sum_{w \geq k} q^{r+1} \cdot \left(\frac{d}{pq}\right)^w \cdot \frac{1}{w!} \cdot \binom{N-2d+w}{\ell} \cdot \binom{w}{k} \\
&\leq p^{2d} \cdot q^{r+1} \cdot \sum_{D_1 \in \text{Supp}(\text{NW}_r)} \sum_{w \geq k} \left(\frac{1}{d^{\alpha-\beta}}\right)^w \cdot \frac{1}{w!} \cdot \binom{N-2d+w}{\ell} \cdot \binom{w}{k}
\end{aligned}$$

The term  $\left(\frac{1}{d^{\alpha-\beta}}\right)^w \cdot \frac{1}{w!} \cdot \binom{N-2d+w}{\ell} \cdot \binom{w}{k}$  is maximized at  $w = k$  as  $\beta < \alpha$ . So,

$$\mathcal{E}[T_1] \leq d \cdot \frac{\gamma^2}{d^{(\alpha-\beta)k} \cdot k!} \cdot \binom{N-2d+k}{\ell}.$$

◀



# A Depth-Five Lower Bound for Iterated Matrix Multiplication\*

Suman K. Bera and Amit Chakrabarti

Department of Computer Science, Dartmouth College  
Hanover, USA  
{suman.k.bera.gr, amit.chakrabarti}@dartmouth.edu

---

## Abstract

We prove that certain instances of the iterated matrix multiplication (IMM) family of polynomials with  $N$  variables and degree  $n$  require  $N^{\Omega(\sqrt{n})}$  gates when expressed as a homogeneous depth-five  $\Sigma\Pi\Sigma\Pi\Sigma$  arithmetic circuit with the bottom fan-in bounded by  $N^{1/2-\epsilon}$ . By a depth-reduction result of Tavenas, this size lower bound is optimal and can be achieved by the weaker class of homogeneous depth-four  $\Sigma\Pi\Sigma\Pi$  circuits.

Our result extends a recent result of Kumar and Saraf, who gave the same  $N^{\Omega(\sqrt{n})}$  lower bound for homogeneous depth-four  $\Sigma\Pi\Sigma\Pi$  circuits computing IMM. It is analogous to a recent result of Kayal and Saha, who gave the same lower bound for homogeneous  $\Sigma\Pi\Sigma\Pi\Sigma$  circuits (over characteristic zero) with bottom fan-in at most  $N^{1-\epsilon}$ , for the harder problem of computing certain polynomials defined by Nisan–Wigderson designs.

**1998 ACM Subject Classification** F.1.3 Complexity Measures and Classes

**Keywords and phrases** arithmetic circuits, iterated matrix multiplication, depth five circuits, lower bound

**Digital Object Identifier** 10.4230/LIPIcs.CCC.2015.183

## 1 Introduction

The fundamental goal of algebraic complexity theory is an understanding of which polynomials can be computed efficiently. Arithmetic formulas and circuits, being the most natural and intuitive model for computing polynomials, are the basis for the notion of *complexity* of a polynomial. They are defined in an analogous way to Boolean formulas and circuits, the key difference being that the *gates* used to build them are addition (+) and multiplication ( $\times$ ) gates, rather than logic gates (more details appear in Section 2).

A classic result in the area is that the symbolic  $n \times n$  determinant – an  $n^2$ -variate polynomial of degree  $n$  – can be computed by a  $\text{poly}(n)$ -sized arithmetic circuit over an arbitrary field [2]. A classic open problem is to prove that the symbolic  $n \times n$  permanent – also  $n^2$ -variate and of degree  $n$  – cannot be so computed. In a highly influential work, Valiant [22] defined complexity classes analogous to P and NP for the algebraic world, which have since come to be called VP (polynomial-sized arithmetic circuits) and VNP (roughly, polynomial-sized arithmetic circuits with a summation quantifier; the permanent has such circuits), and hypothesized that  $\text{VP} \neq \text{VNP}$ . Proving this separation is the preeminent open problem in the area.

Recent work, starting with Agarwal and Vinay [1], has shown that the  $\text{VP} \neq \text{VNP}$  conjecture can be attacked by focusing on *constant-depth* circuits (equivalently, constant-

---

\* This work was partially supported by NSF grant CCF-1217375.



depth formulas). In particular, it suffices to prove certain strong size lower bounds for depth-four  $\Sigma\Pi\Pi\Pi$  circuits: these are layered circuits with four layers of gates, alternating between  $+$ -gates and  $\times$ -gates, with a  $+$ -gate at the top (output) level. A flurry of research over the last two years has greatly advanced our understanding of the power of such circuits. Tavenas [21] has shown that every  $N^{O(1)}$ -sized arithmetic circuit on  $N$  variables computing a polynomial of degree  $n = N^{\Theta(1)}$  can be transformed into a depth-four  $\Sigma\Pi\Pi\Pi$  circuit of size  $N^{O(\sqrt{n})}$  with bottom fan-in at most  $O(\sqrt{n})$ . Moreover the transformation preserves *homogeneity*: if the original circuit is homogeneous – meaning that each  $+$ -gate computes a homogeneous polynomial – then so is the transformed circuit.

In a recent *tour de force*, Kumar and Saraf [16] showed that depth-four homogeneous circuits for the iterated matrix multiplication (IMM) family of polynomials *require*  $N^{\Omega(\sqrt{n})}$  size even without a restriction of the bottom fan-in; again  $N$  and  $n$  represent the number of variables and the degree (respectively), and their proof uses  $N \approx n^{11}$ . Since the IMM polynomials are easily seen to have polynomial-sized arithmetic circuits, this lower bound shows that Tavenas’s depth reduction result is tight in a strong sense.

## 1.1 Our Results

We extend the above Kumar–Saraf theorem to obtain a similar exponential lower bound for depth-five homogeneous  $\Sigma\Pi\Pi\Pi\Sigma$  circuits, albeit with a restriction on the bottom fan-in. Namely, we consider circuits where this bottom fan-in is at most  $N^\mu$ , where  $\mu < 1/2$  is a constant. For each such  $\mu$ , we shall consider a family of  $N$ -variate degree- $n$  IMM polynomials, where  $N = n^{\Theta(q)}$  and  $q$  is a constant depending on  $\mu$ , and show that our restricted depth-five circuits require  $N^{\Omega(\sqrt{n})}$  size to compute these polynomials. By Tavenas’s above theorem, the bound  $N^{\Omega(\sqrt{n})}$  is tight.

The IMM polynomials are defined as follows. The variables are  $\{z_{i,j}^{(h)}\}_{h \in [n], i,j \in [m]}$ , to be thought of as entries of  $m \times m$  matrices  $Z^{(1)}, \dots, Z^{(n)}$ : we use the standard convention that the  $(i, j)$ -entry of a matrix  $A$  is denoted  $a_{i,j}$ . The polynomial  $\text{IMM}_{n,m}$  on these variables is defined as the  $(1, 1)$ -entry of the matrix product  $Z^{(1)}Z^{(2)} \dots Z^{(n)}$ . Thus,

$$\text{IMM}_{n,m} \left( z_{1,1}^{(1)}, \dots, z_{m,m}^{(n)} \right) = \sum_{i_1, i_2, \dots, i_{n-1} \in [m]} z_{1, i_1}^{(1)} z_{i_1, i_2}^{(2)} \dots z_{i_{n-2}, i_{n-1}}^{(n-1)} z_{i_{n-1}, 1}^{(n)}. \quad (1)$$

► **Theorem 1.1 (Main Theorem).** *For every constant  $\mu \in [0, 1/2)$ , there is an integer  $q > 0$  such that the following holds. With  $m = n^q$ , any homogeneous  $\Sigma\Pi\Pi\Pi\Sigma$  circuit with bottom fan-in  $N^\mu$  that computes the  $N$ -variate degree- $n$  polynomial  $\text{IMM}_{n,m}$  has size at least  $N^{\Omega(\sqrt{n})}$ .*

A more precise version of this theorem appears as Theorem 4.2 on page 194.

Proving a super-polynomial lower bound for homogeneous  $\Sigma\Pi\Pi\Pi\Sigma$  circuits was explicitly posed as an open problem by Nisan and Wigderson [18, Section 2.2] in their pioneering work on arithmetic circuit lower bounds. In particular, a depth-five lower bound for IMM was posed as an open problem by Kayal and Saha in a recent work [12] where they obtained such bounds for the so-called Nisan–Wigderson (NW) family of polynomials.<sup>1</sup> Unlike IMM, the NW polynomials are not known to have polynomial-sized arithmetic circuits. This is a strength of our result, because it applies to a potentially “easier” family of polynomials.

Another strength of our result is that, unlike the above Kayal–Saha result, it does not depend on any properties of the underlying field  $\mathbb{F}$  over which the circuit is defined. Their

<sup>1</sup> The name “Nisan–Wigderson” for these polynomials refers to a still earlier work of Nisan and Wigderson [17] that popularized a certain kind of combinatorial design.

result crucially relies on  $\mathbb{F}$  having characteristic zero. Meanwhile a weakness of our result is the  $\mu < 1/2$  requirement; the analogous requirement in Kayal–Saha is that  $\mu < 1$ , which is still a restriction on the structure of the circuit but a weaker one.

We shall prove our depth-five lower bound for a slight restriction of the polynomial in eq. (1) obtained by setting some of its variables to 1 (clearly this only strengthens the result). Our proof will use machinery from the recent work of Kayal and Saha [12] to essentially transform a depth-five circuit into a depth-four one, while controlling the bottom fan-in.

## 1.2 Related Work

In a seminal work, Valiant *et al.* [23] gave the first nontrivial depth-reduction technique for general arithmetic circuits. They proved that a  $\text{poly}(N)$ -sized  $N$ -variate arithmetic circuit that computes a polynomial with degree  $\text{poly}(N)$  can be assumed to be of  $\text{poly}(\log N)$  depth without loss of generality. All subsequent depth-reduction results have built on this work. In particular, Agarwal and Vinay [1] gave a reduction to depth four and the parameters of this reduction were subsequently refined and improved by Koiran [15] and, most recently, by Tavenas [21] who gave the result described earlier.

A consequence of Tavenas’s theorem is that a size lower bound of  $N^{\omega(\sqrt{n})}$  for homogeneous  $\Sigma\Pi\Sigma\Pi$  circuits computing a homogeneous polynomial<sup>2</sup>  $f$  shows that  $f \notin \text{VP}$ ; in fact the circuits can be restricted to a bottom fan-in of  $O(\sqrt{n})$ . In particular, proving such a strong lower bound with  $f$  being either the permanent polynomial or the NW polynomial would imply  $\text{VP} \neq \text{VNP}$ . A number of recent works have pursued this research program and made significant progress.

This research program can be traced back to the groundbreaking work of Nisan and Wigderson [18], which introduced the idea of studying the dimension of the space of partial derivatives of a polynomial  $f$ . Lower bounds on this dimension imply lower bounds on the size of depth-three  $\Sigma\Pi\Sigma$  circuits for  $f$ . In particular, this technique shows that a *homogeneous*  $\Sigma\Pi\Sigma$  circuit computing the  $n \times n$  symbolic determinant (over an arbitrary field) must have size  $2^{\Omega(n)}$ . Gupta *et al.* [7] greatly strengthened this technique by considering “shifted” partial derivatives (see Section 2), and proved that a homogeneous  $\Sigma\Pi\Sigma\Pi$  circuit with bottom fan-in at most  $\sqrt{n}$  that computes either the  $n \times n$  determinant or the  $n \times n$  permanent must have size  $2^{\Omega(\sqrt{n})}$ . Kayal *et al.* [13] then proved a larger lower bound of  $N^{\Omega(\sqrt{n})}$  for the same class of circuits, for the “harder” problem of computing an  $N$ -variate degree- $n$  NW polynomial. Fournier *et al.* [4] proved the same lower bound for the problem of computing certain IMM polynomials.

The next major conceptual advance was made by Kayal *et al.* [11], who further strengthened the partial derivatives technique by adding a multilinear projection step, arriving at the “dimension of projected shifted partials” measure. Using this, and further applying well-chosen random restrictions, they removed the bottom fan-in restriction and gave an  $N^{\Omega(\sqrt{n})}$  lower bound for homogeneous  $\Sigma\Pi\Sigma\Pi$  circuits computing NW polynomials. However, their proof introduced a new restriction: the underlying field had to have characteristic zero. The aforementioned recent work by Kumar and Saraf [16] does not have such a restriction on the field and gives the same  $N^{\Omega(\sqrt{n})}$  lower bound for certain NW polynomials as well as certain IMM polynomials. Since  $\text{IMM} \in \text{VP}$ , this proves the tightness of Tavenas’s theorem [21].

<sup>2</sup> Strictly speaking, one considers the complexity not of a single polynomial but of a family of polynomials  $\{f_N\}_{N \in \mathcal{I}}$  for some infinite index set  $\mathcal{I} \subseteq \mathbb{N}$ .

Along different lines Grigoriev and Karpinski [5], and Grigoriev and Razborov [6] considered (not necessarily homogeneous)  $\Sigma\Pi\Sigma$  circuits over a finite field  $\mathbb{F}$  and proved that computing the  $\text{MOD}_q$  function on  $n$  variables, where  $q \neq \text{char}(\mathbb{F})$  is a prime, requires size  $2^{\Omega(n)}$ . In contrast, over a field of characteristic zero, a result of Gupta *et al.* [8] shows that a polynomial-sized  $N$ -variate arithmetic circuit can be converted to a non-homogeneous  $\Sigma\Pi\Sigma$  circuit of size  $N^{O(\sqrt{n})}$ . Thus, another approach to proving  $\text{VP} \neq \text{VNP}$  would be to show strong enough lower bounds for general  $\Sigma\Pi\Sigma$  circuits.

Recently Kayal and Saha [12] proved that a  $\Sigma\Pi\Sigma$  circuit over a field of characteristic zero computing certain  $N$ -variate degree- $n$  polynomials – namely, NW and IMM polynomials with  $N = n^{\Theta(1)}$  – must have size  $N^{\Omega(\sqrt{n})}$ , provided the bottom fan-in is at most  $\sqrt{n}$ . Their technique involves converting the  $\Sigma\Pi\Sigma$  circuit into a homogeneous  $\Sigma\Pi\Sigma\Pi\Sigma$  circuit with bounded bottom fan-in (precisely the class of circuits that this paper is about) and then proving lower bounds for the resulting depth-five circuits. In fact their depth-five circuits have a very special structure, which they then exploit to obtain their NW and IMM results. Without using this special structure, they are still able to obtain lower bounds for NW, but not IMM.

As noted in Theorem 1.1, this paper gives such a depth-5 lower bound for IMM. Our own proof builds on the ideas of Kayal and Saha.

The excellent survey by Shpilka and Yehudayoff [20] gives a much more detailed overview of classic and modern results on arithmetic circuits. Two new surveys by Kayal and Saptharishi [14], and Saptharishi [19] cover recent progress on constant-depth lower bounds.

## 2 Preliminaries and Proof Outline

All arithmetic circuits studied in this paper will be constant-depth, layered, and homogeneous, with gates of arbitrary fan-in except where noted. A layer is either a  $\Sigma$ -layer (consisting of  $+$ -gates only) or a  $\Pi$ -layer (consisting of  $\times$ -gates only). The output layer always consists of a single  $+$ -gate. Notation of the form  $\Sigma\Pi\Sigma\Pi$  indicates the number and types of layers with the leftmost symbol corresponding to the output (a.k.a. top) layer and the rightmost symbol corresponding to the bottom layer, whose gates read only input variables.<sup>3</sup> Wires feeding  $+$ -gates are labeled with coefficients from the underlying field  $\mathbb{F}$ : thus a  $+$ -gate computes an arbitrary linear form over  $\mathbb{F}$ .

Following Kumar and Saraf [16], we shall consider the following restriction of the IMM polynomial defined in eq. (1). Let

$$m = n^q, \quad n = (B + 2)k, \quad (2)$$

for some integers  $q, B$ , and  $k$ . Eventually, we shall take  $B = \Theta(\sqrt{n})$ ,  $k = \Theta(\sqrt{n})$ , and  $q$  large enough but constant. We partition the sequence of matrices  $Z^{(1)}, \dots, Z^{(n)}$  into  $k$  contiguous subsequences, which we call *blocks*. In the  $h$ th block, we denote the first matrix as  $Y^{(h)}$ , the next  $B$  matrices as  $X^{(h,1)}, X^{(h,2)}, \dots, X^{(h,B)}$ , and the last matrix as  $J^{(h)}$ . We then set all entries of each  $J^{(h)}$  to be 1. We shall denote the resulting polynomial, which is slightly smaller than the original and uses a different set of variable names, as

$$f_{n,q}(x_{1,1}^{(1,1)}, \dots, x_{m,m}^{(k,B)}, y_{1,1}^{(1)}, \dots, y_{m,m}^{(k)}). \quad (3)$$

Clearly,  $\deg f_{n,q} \leq n$  and  $f_{n,q}$  is  $N$ -variate for  $N = m^2(n - k)$ .

<sup>3</sup> When studying non-homogeneous circuits, we must also allow the bottom layer gates to read the constant 1.

Our lower bound is based on the complexity measure termed “dimension of projected shifted partials” (DPSP), whose history we have recounted in Section 1.2. We now define the DPSP measure. Fix a field  $\mathbb{F}$  and a set of variables  $x_1, \dots, x_N$ . Consider a polynomial  $f(x_1, \dots, x_N) \in \mathbb{F}[x_1, \dots, x_N]$ . Let  $\alpha = x_{i_1} \dots x_{i_k}$  be a multilinear monomial in the same variables. We use the compact notation  $\partial_\alpha f := \partial^k f / \partial x_{i_1} \dots \partial x_{i_k}$ , calling it the partial derivative of  $f$  with respect to  $\alpha$ . Let  $\mathcal{M}$  be a set of multilinear monomials and  $\ell \geq 0$  be an integer. We define

$$\text{DPSP}_{\mathcal{M}, \ell}(f) := \dim \text{span proj shift}_\ell \{ \partial_\alpha f : \alpha \in \mathcal{M} \}, \tag{4}$$

where  $\text{shift}_\ell f = \{ \beta f : \beta \text{ is a monomial of degree } \ell \}$ ,  $\text{proj } f$  is the projection of  $f$  onto the subspace of the  $\mathbb{F}$ -vector-space  $\mathbb{F}[x_1, \dots, x_N]$  spanned by multilinear monomials, and these operators are extended to sets of polynomials in the natural way.

For fixed choices of  $\mathcal{M}$  and  $\ell$ , the measure  $\text{DPSP}_{\mathcal{M}, \ell}$  is easily seen to be subadditive. It is a good complexity measure because it can be nontrivially upper-bounded for “simple” circuits. Let us call a circuit  $t$ -supported if at most  $t$  distinct variables feed each bottom-level gate.

► **Lemma 2.1** (Essentially [10, Corollary 12]). *Let  $C$  be a  $t$ -supported degree- $n$  homogeneous  $\Sigma\Pi\Sigma\Pi$  circuit on  $N$  variables, with top fan-in at most  $S_0$ . Let  $\mathcal{M}$  be a set of degree- $k$  multilinear monomials on these  $N$  variables and let  $\ell \geq 0$  be an integer such that  $\ell + kt \leq N/2$ . Then*

$$\text{DPSP}_{\mathcal{M}, \ell}(C) \leq S_0 \binom{2n/t + 1}{k} \binom{N}{\ell + kt}. \quad \blacktriangleleft$$

To apply this to depth five circuits with small support, we proceed as in Kayal and Saha [12]: we perform a random restriction. That is, we kill (set to zero) all variables  $x_i$  lying outside a suitably randomly chosen subset  $V$ . This will simplify a polynomial  $f$  to a “smaller” polynomial, which we will denote  $f|_V$ . The crux of our argument is to show that a sufficiently strong restriction will, w.h.p., simplify a depth-five circuit into a depth-four circuit (the truth is a little more subtle; see Lemma 2.4). At the same time, we do not want to apply too strong a restriction, for otherwise the IMM polynomial itself might simplify too much. We desire that, w.h.p., the restricted polynomial  $f_{n,q}|_V$  (see eq. (3)) still has “high” complexity, with respect to our DPSP measure.

### 2.1 Random Restrictions and Their Effect on IMM

Let  $\mathcal{V}_{n,q}$  denote the set of variables of the polynomial  $f_{n,q}$ ; see eq. (3). We now define a distribution over subsets of  $\mathcal{V}_{n,q}$  by describing a procedure for sampling a random subset,  $V$ . The set  $V$  is a union (over  $h, h'$ , and  $i$ ) of random subsets  $V_i^{(h, h')}$  and  $V_i^{(h)}$ , which are subsets of the variables in the  $i$ th row of  $X^{(h, h')}$  and  $Y^{(h)}$  respectively; these subsets are mutually independent. Each such subset is chosen uniformly conditioned on its size being some particular quantity, as follows (the parameters  $b$  and  $\lambda$  will be fixed later).

- For each  $h$ ,  $|V_1^{(h)}| = m^b = n^{bq}$ , where  $b \in (0, 1)$ . Further,  $|V_i^{(h)}| = 0$  for  $i \neq 1$ .
- For each  $h$ ,  $|V_i^{(h, 1)}| = n^\lambda$  for each  $i$ , where  $\lambda \approx 2$ .
- For each  $h$  and  $h'$ , with  $2 \leq h' \leq B - 2 \log n$ ,  $|V_i^{(h, h')}| = 2$  for each  $i$ .
- For each  $h$  and  $h'$ , with  $h' > B - 2 \log n$ ,  $|V_i^{(h, h')}| = 1$  for each  $i$ .

Then, as mentioned above, we set

$$V := \bigcup_{i=1}^m \bigcup_{h=1}^k \left( V_i^{(h)} \cup \bigcup_{h'=1}^B V_i^{(h, h')} \right). \tag{5}$$

Technically, our proof is all about studying the effects of restricting our depth-five circuits and the IMM polynomial to this random set  $V$ . This random restriction is a small generalization of the one used by Kumar and Saraf [16], where we have introduced  $b$  as a tunable parameter. Therefore, their (highly technical) analysis of the effect of this random restriction on the IMM polynomial largely carries over. We shall now explain the final outcome of this analysis.

To this end, we introduce the following key parameters:

$$k := 32\sqrt{n}; \quad (\text{this then determines } B) \quad (6)$$

$$\hat{n} := Bk = n - 2k; \quad (\text{the number of } X \text{ matrices}) \quad (7)$$

$$\ell := \frac{N}{2} \left( 1 - \frac{\ln n}{\Gamma\sqrt{n}} \right), \quad \text{where} \quad (8)$$

$$\Gamma := 2 + o(1) \quad \text{is chosen such that } n^{\sqrt{n}} \left( \frac{N}{N-\ell} \right)^{\hat{n}} = \left( \frac{N}{\ell} \right)^{\hat{n}}; \quad (9)$$

$$\lambda := 2 - \frac{1 + o(1)}{32\Gamma} \quad \text{is chosen such that } n^{\lambda k} \cdot 2^{\hat{n} - (1+2\log n)k} = \left( \frac{N}{N-\ell} \right)^{\hat{n}}. \quad (10)$$

For each  $V$  drawn as indicated above, let  $\mathcal{M}(V)$  denote the set of all monomials obtained by picking exactly one  $y$ -variable from each set  $V_1^{(h)}$ ; the degree of each such monomial is then  $k$ .

► **Lemma 2.2** (Slight generalization of [16, Lemma 8.1]). *Suppose  $bq > 1$ . Then, for every realization of the random set  $V$ , there exists  $\mathcal{M}'(V) \subseteq \mathcal{M}(V)$  such that  $|\mathcal{M}'(V)| = n^{\sqrt{n}}$  and  $\forall \alpha_1 \neq \alpha_2 \in \mathcal{M}'(V)$ ,*

$$|\text{supp}(\alpha_1) \setminus \text{supp}(\alpha_2)| = |\text{supp}(\alpha_2) \setminus \text{supp}(\alpha_1)| \geq k - \sqrt{n},$$

where the support  $\text{supp}(\alpha)$  of a monomial  $\alpha$  is defined as the set of variables that appear in  $\alpha$ .

► **Lemma 2.3** (Essentially [16, Lemma 8.9]). *Suppose  $bq > 1$ . With probability at least 0.9, the above set  $\mathcal{M}'(V)$  contains a subset  $\mathcal{M}''(V)$  such that*

$$\text{DPSP}_{\mathcal{M}''(V), \ell}(f_{n,q}|_V) \geq \frac{n^{\sqrt{n}}}{O(n^{\sqrt{n}/8}) \cdot n^{o(\sqrt{n})}} \left( \frac{N}{N-\ell} \right)^{\hat{n}} \binom{N-\hat{n}}{\ell}.$$

## 2.2 Circuit Decomposition Under Random Restrictions

To prove our depth-five lower bound using the DPSP lower bound given by Lemma 2.3, we will need to extend Lemma 2.1 as discussed right after its statement. We will analyze the random restriction defined in Section 2.1 to establish the following decomposition lemma.

► **Lemma 2.4** (Analogous to [12, Lemma 11]). *For each constant  $\mu < 1/2$ , there exists an integer  $q = q(\mu)$  such that the following holds. Let  $C$  be an  $N^\mu$ -supported degree- $n$  homogeneous  $\Sigma\Pi\Sigma\Pi\Sigma$  circuit on the variables  $\mathcal{V}_{n,q}$ , with size  $S \leq n^{\varepsilon\sqrt{n}}$  for some small positive constant  $\varepsilon$ . Let the random set  $V$  be drawn as above, with  $b$  chosen such that  $bq \geq \lambda$ . Then with probability  $1 - o(1)$ ,*

$$C|_V = C' + g, \quad (11)$$

where  $C'$  is a  $(\sqrt{n}/64)$ -supported degree- $n$  homogeneous  $\Sigma\Pi\Sigma\Pi$  circuit with top fan-in at most that of  $C$ , and  $g$  is a polynomial each of whose monomials has a variable raised to the third or higher power.

The proof of the above lemma is our main technical contribution. It occupies most of Section 3.

The projection step in the definition of DPSP ensures that the polynomial  $g$  in eq. (11) satisfies  $\text{DPSP}_{\mathcal{M},\ell}(g) = 0$  for every choice of  $\mathcal{M}$  and  $\ell$ . Furthermore, the bound on the bottom fan-in of  $C'$  enables us to apply Lemma 2.1. Recalling that DPSP is a subadditive measure, we then obtain the following upper bound (setting  $t = \sqrt{n}/64$  in Lemma 2.1).

► **Lemma 2.5** (Analogous to [12, Lemma 9]). *Let  $\mu, q, C, S$ , and  $V$  be as in the previous lemma. Then the following event occurs with probability  $1 - o(1)$ . For all sets  $\mathcal{M}$  of degree- $k$  multilinear monomials and all  $\ell \geq 0$  such that  $\ell + k\sqrt{n}/64 \leq N/2$ , we have*

$$\text{DPSP}_{\mathcal{M},\ell}(C|_V) \leq S \binom{128\sqrt{n} + 1}{k} \binom{N}{\ell + k\sqrt{n}/64}. \quad \blacktriangleleft$$

Our final lower bound – Theorem 1.1 – then follows by combining Theorems 2.3 and 2.5 and using the parameter settings in eqs. (6)–(10).

### 3 Proof Details

Lemmas 2.1 and 2.3 are essentially restatements of the corresponding lemmas from previous works [10, 16]. It remains to prove Lemma 2.2 and 2.4.

#### 3.1 A Well-Spaced Collection of Derivatives

We prove the first of these lemmas, which guarantees that the set  $\mathcal{M}(V)$  of monomials with respect to which we shall be taking derivatives contains a large set of pairwise far monomials.

**Proof of Lemma 2.2.** Recall that  $|V_1^{(h)}| = n^{bq}$  for each  $h \in [k]$ . Therefore  $\mathcal{M}(V)$  maps bijectively to  $V_1^{(1)} \times \cdots \times V_1^{(k)}$  in a natural way and thence to  $[n^{bq}]^k$  in an artificial way. Let  $\mathbb{K}$  be the largest finite field whose order is at most  $n^{bq}$ ; note that  $|\mathbb{K}| \geq n^{bq}/2$ . Then  $\mathbb{K}^k$  maps injectively (artificially) into  $\mathcal{M}(V)$ , via an injection  $\iota$ , say.

Consider a Reed–Solomon code  $\mathcal{C} \subseteq \mathbb{K}^k$  where the codewords are evaluations of polynomials in  $\mathbb{K}[w]$  of degree at most  $\sqrt{n}$  at  $k$  distinct points in  $\mathbb{K}$ . Then

$$|\mathcal{C}| = |\mathbb{K}|^{\sqrt{n}+1} \geq (n^{bq}/2)^{\sqrt{n}+1} \geq n^{\sqrt{n}},$$

since  $bq > 1$ . Pick  $\mathcal{M}'(V)$  to be an arbitrary  $n^{\sqrt{n}}$ -sized subset of  $\iota(\mathcal{C})$ . The code  $\mathcal{C}$ , by construction, has Hamming distance at least  $k - \sqrt{n}$ . This directly translates to the desired monomial distance property for  $\mathcal{M}'(V)$ . ◀

**Proof of Lemma 2.3.** As we noted while stating this lemma, it essentially restates Lemma 8.9 from Kumar and Saraf [16]. The main concern is that for small  $b$  our set  $\mathcal{M}'(V)$  above could be much smaller than their corresponding set. However, an examination of the proof of their Lemma 8.9 shows that the only properties of  $\mathcal{M}'(V)$  that are needed are the size bound  $|\mathcal{M}'(V)| \geq n^{\sqrt{n}}$  and the above farness property, both of which our Lemma 2.2 guarantees. ◀

#### 3.2 The Main Lemma: Circuit Decomposition

We prove the remaining lemma which establishes the circuit decomposition indicated by eq. (11). The following technical lemma will be useful in the analysis.



► **Lemma 3.1.** *Given integers  $0 < t \leq s' \leq s$  and sets  $A$  and  $B$  with  $|A| = s$ ,  $|B| = t$ , and  $B \subseteq A$ , let  $R$  be a random subset of  $A$  chosen uniformly conditioned on  $|R| = s'$ . Then  $\Pr[B \subseteq R] \leq (s'/s)^t$ . ◀*

To start the proof of Lemma 2.4, consider  $C$ , an arbitrary  $N^\mu$ -supported degree- $n$  homogeneous  $\Sigma\Pi\Sigma\Pi\Sigma$  circuit with size  $S \leq n^{\varepsilon\sqrt{n}}$ , on the variables  $\mathcal{V}_{n,q}$ , that computes the IMM polynomial  $f_{n,q}$ . Fix this  $C$  for the rest of this section. Expanding  $C$  into a formula, we have

$$C = \sum_i \prod_j \sum_r Q_{ijr}, \quad (12)$$

where each  $Q_{ijr}$  is a product of linear forms, each such linear form having at most  $N^\mu$  variables. The proof now splits into two cases: the *thin case*, when the bottom fan-in is below  $N^{1/4}$  and the *fat case*, when the bottom fan-in is  $N^{1/4}$  or more.

### 3.3 The Thin Case

We consider the case when  $0 \leq \mu < \frac{1}{4}$ .

Let the random set  $V$  be drawn as described in Section 2.1. A monomial survives the restriction to  $V$  iff all its variables belong to  $V$ . Now Lemma 3.1 implies the following bounds for a monomial  $\alpha$  with  $|\text{supp}(\alpha)| = t = O(\sqrt{n})$ .

- For each  $h$ , if  $\alpha$  has variables only from the first row of  $Y^{(h)}$ , then its survival probability is at most  $m^{-(1-b)t} = n^{bqt}n^{-qt}$ .
- For each  $h$ , if  $\alpha$  has variables only from the  $i$ th row of  $X^{(h,1)}$ , then its survival probability is at most  $n^{-(q-\lambda)t} = n^{\lambda t}n^{-qt}$ .
- For each  $h, h'$  and  $i$ , with  $2 \leq h' \leq B - 2 \log n$ , if  $\alpha$  has variables only from the  $i$ th row of  $X^{(h,h')}$ , then its survival probability is at most  $(2/m)^t = 2^t n^{-qt} = n^{t/\log n} n^{-qt}$ .
- For each  $h, h'$  and  $i$ , with  $h' > B - 2 \log n$ , if  $\alpha$  has variables only from the  $i$ th row of  $X^{(h,h')}$ , then its survival probability is at most  $m^{-t} = n^{-qt}$ .

The hypotheses of Lemma 2.4 include the condition  $bq \geq \lambda$ , and eq. (10) implies  $\lambda > 1$ . Therefore the largest of these bounds is the first one, i.e.,  $n^{-(1-b)qt}$ .

Since all the random subsets mentioned above are mutually independent, even if  $\alpha$ 's variables are spread out arbitrarily among multiple rows of multiple matrices, its survival probability is still at most  $n^{-(1-b)qt}$ .

Let  $C|_V = \sum_i \prod_j \sum_r Q'_{ijr}$  where  $Q'_{ijr}$  is a product of linear forms. Assume for some  $(i, j, r)$  that  $\deg(Q'_{ijr}) = 2t$ ; if  $\deg(Q'_{ijk}) > 2t$ , then we only consider the product of the “first”  $2t$  linear forms. Then the number of monomials in  $Q'_{ijk}$  is at most  $(N^\mu)^{2t}$ . Consider the *bad monomials* in  $Q'_{ijk}$ , defined as ones where each variable has degree at most 2. These monomials have support at least  $t$ ; the event that one of them survives is a *bad event*. If not a single bad monomial survives, then the circuit  $C$  decomposes into two circuits: a  $2t$ -supported degree- $n$  homogeneous  $\Sigma\Pi\Sigma\Pi$  circuit  $C'$  with top fan-in at most that of  $C$ , and a circuit  $g$  wherein each monomial has a variable raised to the third or higher power. Setting  $t = \sqrt{n}/128$ , this is exactly the decomposition we seek.

It remains to prove that the above bad event has probability  $o(1)$ . The probability that a bad monomial survives the random restriction is at most  $n^{-(1-b)qt}$ , as noted above. By a union bound, the bad event has probability at most

$$S n^{-(1-b)qt} (N^\mu)^{2t} \leq n^{\varepsilon\sqrt{n}} n^{-(1-b)qt} n^{(2q+1)2\mu t} = \left( n^{128\varepsilon - (1-b)q + (2q+1)2\mu} \right)^t.$$



Since  $t = \Theta(\sqrt{n})$ , we can bound this by  $n^{-\Omega(\sqrt{n})}$  by ensuring

$$128\varepsilon - (1 - b)q + (2q + 1)2\mu < 0. \quad (13)$$

Clearly it suffices to ensure that

$$(1 - b)q = (2q + 1)2\mu + 129\varepsilon. \quad (14)$$

Recall that we want  $bq \geq \lambda \approx 2$  and  $b \in (0, 1)$ . So we need  $(1 - b)q = q - bq \leq q - \lambda$ , i.e.,

$$(2q + 1)2\mu + 129\varepsilon \leq q - \lambda \iff q \geq \frac{\lambda + 129\varepsilon + 2\mu}{1 - 4\mu}, \quad (15)$$

where we have used  $\mu < 1/4$ .

We set  $q$  to be the smallest integer satisfying (15), then set  $b$  to satisfy (14). Then we do have  $bq \geq \lambda$  as well as  $b \in (0, 1)$  as required.

### 3.4 The Fat Case

We consider the remaining case, when  $\frac{1}{4} \leq \mu < \frac{1}{2}$ .

We imagine the random restriction as being performed in two phases. Phase 1 chooses “large” random subsets of each row of each matrix in the IMM polynomial (for the  $Y$ -matrices, only the first row is used). Then Phase 2 chooses smaller random subsets, of the desired target sizes as in Section 2.1. The net effect is the same as the random restriction described in Section 2.1.

#### Phase 1

Let  $a$  be a parameter such that  $0 < b < a < 1$ ; its value will be fixed in the later analysis.

We now define a distribution over subsets of  $\mathcal{V}_{n,q}$  for sampling a random subset,  $W$ . Similar to  $V$ ,  $W$  is also a union of random subsets  $W_1^{(h)}$  and  $W_i^{(h,h')}$  over  $h, h'$  and  $i$ , where  $W_i^{(h)}$  and  $W_i^{(h,h')}$  are subsets of variables in the  $i$ th row of  $Y^{(h)}$  and  $X^{(h,h')}$  respectively; these subsets are mutually independent. Each subset is chosen uniformly conditioned on its size being  $m^a$ . In the first phase, we consider a restriction to  $W$ , i.e., all variables outside  $W$  are set to zero.

Consider the probability that a monomial  $\alpha$ , with  $|\text{supp}(\alpha)| = t = O(\sqrt{n})$ , survives Phase 1. By Lemma 3.1, if  $\alpha$ 's variables come only from the first row of  $Y^{(h)}$  for some particular  $h$ , or only from the  $i$ th row of some particular  $X^{(h,h')}$ , then its survival probability is at most  $m^{-(1-a)t} = n^{-(1-a)qt}$ . Since all the random subsets are mutually independent, even if  $\alpha$ 's variables are spread out arbitrarily among multiple rows of multiple matrices, its survival probability is still at most  $n^{-(1-a)qt}$ .

#### Phase 2

In this phase, we sample  $V_1^{(h)} \subseteq W_1^{(h)}$  and  $V_i^{(h,h')} \subseteq W_i^{(h,h')}$ , uniformly and independently, subject to the cardinality constraints given in Section 2.1. We then define  $V$  as in eq. (5).

Let  $\alpha$  be a monomial with  $|\text{supp}(\alpha)| = t = O(\sqrt{n})$ . If the variables in  $\alpha$  all come from a single set  $W_1^{(h)}$  or  $W_i^{(h,h')}$ , then we can bound the probability of  $\alpha$  surviving this second phase exactly as in Section 3.3, by using Lemma 3.1.

- If the variables come from  $W_1^{(h)}$ , the survival probability is at most  $m^{-(a-b)t} = n^{bqt}n^{-aqt}$ .
- If the variables come from  $W_i^{(h,h')}$ , the survival probability is at most  $n^{-(a-q-\lambda)t} = n^{\lambda t}n^{-aqt}$ .

- If the variables come from  $W_i^{(h,h')}$ , where  $2 \leq h' \leq B - 2 \log n$ , then the survival probability is at most  $(2/m^{-a})^t = 2^t n^{-aqt} = n^{t/\log n} n^{-aqt}$ .
- If the variables come from  $W_i^{(h,h')}$ , where  $h' > B - 2 \log n$ , then the survival probability is at most  $m^{-at} = n^{-aqt}$ .

Again, recalling that  $bq \geq \lambda > 1$ , we see that the largest of these bounds is the first one, i.e.,  $n^{-(a-b)qt}$ . Since all the random subsets mentioned above are mutually independent, even if  $\alpha$ 's variables are spread out arbitrarily among multiple rows of multiple matrices, its survival probability in phase 2 is still at most  $n^{-(a-b)qt}$ .

### Effect of Phase 1 Restriction

We now analyze the effect of the phase 1 random restriction on the circuit  $C$ . Recall the expansion in eq. (12). Let  $Q_{ijr} = \prod_u L_u$  where each  $L_u$  is a linear form with (w.l.o.g.) exactly  $N^\mu$  terms.

Observe that the survival probability of each variable in  $C$  is at most  $n^{-(1-a)q}$ . Therefore, by linearity of expectation,

$$\mathbb{E}[\text{number of surviving terms in } L_u|_W] \leq N^\mu n^{-(1-a)q} \leq n^{(2q+1)\mu - (1-a)q} =: T. \quad (16)$$

We would like to bound the probability that the bottom fan-in of  $C|_W$  greatly exceeds this bound  $T$ . This is not a straightforward Chernoff bound because the number of surviving terms in  $L_u|_W$  is a sum of *dependent* indicator random variables. However, the dependency is of a benign sort. To see this, we recall some facts from probability theory, proved in, e.g., [3, Section 3.1] and [9].

► **Fact 3.2.** *Negative association of random variables is closed under products. That is, if  $X_1, \dots, X_n$  and  $Y_1, \dots, Y_m$  are two independent collections of random variables that are separately negatively associated, then the union  $X_1, \dots, X_n, Y_1, \dots, Y_m$  is also negatively associated.*

► **Fact 3.3.** *Let a subset  $R \subseteq [n]$  be drawn uniformly at random, conditioned on  $|R| = k$ , for some  $k \leq n$ , and let  $X_i$  be an indicator for the event  $i \in R$ . Then the collection  $X_1, \dots, X_n$  is negatively associated.*

► **Fact 3.4.** *The Chernoff–Hoeffding bounds apply as is to a sum of negatively associated random variables.*

Using these facts, we see that standard Chernoff bounds may be applied to the number of surviving terms in  $L_u|_W$ . Doing so and applying a union bound over all linear forms  $L_u$  gives us

$$\Pr [\text{bottom fan-in of } C|_W \geq (1 + \sqrt{3})T] \leq Se^{-T} \leq n^{\varepsilon\sqrt{n}} e^{-T}.$$

For this probability to be  $o(1)$ , it suffices to have

$$\begin{aligned} \varepsilon\sqrt{n} \ln n - n^{(2q+1)\mu - (1-a)q} &\leq -\omega(1) \quad (\text{using eq. (16)}) \\ \Leftrightarrow (2q+1)\mu - (1-a)q &\geq \frac{1}{2} + \Theta(1). \end{aligned} \quad (17)$$

### Effect of Phase 2 Restriction

After phase 1, with high probability the bottom fan-in of  $C|_W$  is bounded by  $(1 + \sqrt{3})T$ . Assuming that this bound holds, we analyze the effect of phase 2 on  $C|_W$ . This analysis is analogous to that in the thin case.

Let  $C|_W = \sum_i \prod_j \sum_k Q'_{ijk}$  where  $Q'_{ijk}$  is a product of linear forms. Assume for some  $i, j, k$ , that  $\deg(Q'_{ijk}) = 2t$ ; if  $\deg(Q'_{ijk}) > 2t$ , then we only consider the product of the first  $2t$  linear forms. Then the number of monomials in  $Q'_{ijk}$  is at most  $(1 + \sqrt{3})^{2t} T^{2t}$ . Consider the *bad monomials* in  $Q'_{ijk}$ : those where each variable has degree at most 2. By our previous analysis, the probability that such a monomial survives phase 2 is at most by  $n^{-(a-b)qt}$ . If no bad monomial survives, then  $C|_W$  indeed decomposes into two circuits as desired: a  $2t$ -supported degree- $n$  homogeneous  $\Sigma\Pi\Sigma\Pi$  circuit  $C'$  with top fan-in at most that of  $C$  and a circuit  $g$  wherein each monomial has a variable raised to a power  $\geq 3$ . We set  $t = \sqrt{n}/128$  to obtain the decomposition we seek.

By a union bound over the at most  $S$  bad monomials, the probability that no bad monomial survives phase 2 – which we would like to bound by  $o(1)$  – is at most

$$\begin{aligned} S n^{-(a-b)qt} (1 + \sqrt{3})^{2t} T^{2t} &\leq n^{\varepsilon\sqrt{n}} n^{-(a-b)qt} (1 + \sqrt{3})^{2t} (n^{(2q+1)\mu - (1-a)q})^{2t} \\ &\leq (n^{128\varepsilon - (a-b)q - 2(1-a)q + 2(2q+1)\mu})^t (1 + \sqrt{3})^{2t}. \end{aligned}$$

Since  $t = \Theta(\sqrt{n})$ , we can bound this by  $n^{-\Omega(\sqrt{n})}$  by ensuring that

$$128\varepsilon - (a-b)q - 2(1-a)q + 2(2q+1)\mu < 0. \quad (18)$$

Recall that we also want  $a, b, q$  to satisfy  $bq \geq \lambda \approx 2$  as well as the phase 1 condition (17). Moreover, for the two-phase random restriction process to make sense, we want  $0 < b < a < 1$ . We claim that it suffices to choose  $a, b$ , and  $q$  such that

$$(1-a)q = (2q+1)\mu - 0.51, \quad \text{and} \quad (19)$$

$$(a-b)q = 2 \times 0.51 + 129\varepsilon. \quad (20)$$

Clearly condition (17) is satisfied. By adding (19) and  $2 \times (20)$ , we see that condition (18) is also satisfied. We will soon set  $q$  to a positive integer, satisfying  $b < a$ . By adding (19) and (20), we get

$$(2q+1)\mu + 0.51 + 129\varepsilon = (1-b)q. \quad (21)$$

We want  $(1-b)q = q - bq \leq q - \lambda$ , i.e.,

$$(2q+1)\mu + 0.51 + 129\varepsilon \leq q - \lambda \iff q \geq \frac{\lambda + 0.51 + 129\varepsilon + \mu}{1 - 2\mu}, \quad (22)$$

where we used  $\mu < 1/2$ . We set  $q$  to be the smallest integer satisfying condition (22). Next we set  $a$  and  $b$  satisfying eq. (19) and eq. (20) respectively. Now we want  $a < 1$ , or equivalently

$$(2q+1)\mu - 0.51 > 0 \iff q > \frac{1}{2} \left( \frac{0.51}{\mu} - 1 \right). \quad (23)$$

So we want

$$\begin{aligned} \frac{1}{2} \left( \frac{0.51}{\mu} - 1 \right) &< \frac{\lambda + 0.51 + 129\varepsilon + \mu}{1 - 2\mu} \\ \iff (\lambda + 0.51 + 129\varepsilon)2\mu + 2\mu^2 &> (1 - 2\mu)(0.51 - \mu) \\ \iff (\lambda + 0.51 + 129\varepsilon)2\mu + 2\mu^2 &> 0.51 - 2.02\mu + 2\mu^2 \\ \iff \mu &> \frac{0.51}{2(\lambda + 0.51 + 129\varepsilon) + 2.02}. \end{aligned}$$

Since,  $1 < \lambda < 2$  and  $\mu \geq 1/4$ , the above inequality does hold.

This completes the proof of Lemma 2.4.

#### 4 Final Result and Discussion

We now put together the lemmas proven so far to obtain our final lower bound on homogeneous  $N^\mu$ -supported  $\Sigma\Pi\Sigma\Pi\Sigma$  circuits for IMM.

The following estimation will be useful in our calculations.

► **Lemma 4.1** (See, e.g., [7, Lemma 6]). *Let  $a(n), f(n), g(n) : \mathbb{Z}_{>0} \rightarrow \mathbb{Z}_{>0}$  be integer valued functions such that  $f + g = o(a)$ . Then*

$$\frac{(a+f)!}{(a-g)!} = a^{f+g} \cdot e^{\pm O((f+g)^2/a)}. \quad \blacktriangleleft$$

With this, we are ready to prove our result.

► **Theorem 4.2** (Precise version of Main Theorem 1.1). *For every constant  $\mu \in [0, 1/2)$ , there is an integer  $q > 0$  such that the following holds. Let  $C$  be a homogeneous  $N^\mu$ -supported  $\Sigma\Pi\Sigma\Pi\Sigma$  circuit that computes the  $N$ -variate degree- $n$  IMM polynomial  $f_{n,q}$  mentioned in Equation (3). Then  $C$  has size at least  $N^{\Omega(\sqrt{n})}$ .*

**Proof.** Suppose  $C$  has size  $S$ . Clearly we may choose an arbitrary small constant  $\varepsilon > 0$  and proceed under the assumption that  $S \leq n^{\varepsilon\sqrt{n}}$ . So we make this assumption.

Let  $V$  be a random subset of  $\mathcal{V}_{n,q}$ , the variable set of  $f_{n,q}$ , sampled according to the distribution described in Section 2.1. By Lemma 2.5, for all sets  $\mathcal{M}$  of degree- $k$  multilinear monomials and all  $\ell \geq 0$  such that  $\ell + k\sqrt{n}/64 \leq N/2$ ,

$$\text{DPSP}_{\mathcal{M},\ell}(C|_V) \leq S \binom{128\sqrt{n}+1}{k} \binom{N}{\ell + k\sqrt{n}/64}$$

with probability  $1 - o(1)$ .

By Lemma 2.3, with probability at least 0.9 there exists a set  $\mathcal{M}''(V)$  of degree- $k$  multilinear monomials such that

$$\text{DPSP}_{\mathcal{M}''(V),\ell}(f_{n,q}|_V) \geq \frac{n^{\sqrt{n}}}{O(n^{\sqrt{n}/8}) \cdot n^{o(\sqrt{n})}} \left(\frac{N}{N-\ell}\right)^{\hat{n}} \binom{N-\hat{n}}{\ell}.$$

for all  $\ell > 0$ . Hence with non-zero probability both these bounds hold. Comparing the above two bounds, and using parameters  $k = 32\sqrt{n}$  and  $\ell = \frac{N}{2} \left(1 - \frac{\ln n}{\Gamma\sqrt{n}}\right)$  from eq. (6) and (8), we get

$$\begin{aligned} S &\geq \frac{\frac{n^{\sqrt{n}}}{O(n^{\sqrt{n}/8}) \cdot n^{o(\sqrt{n})}} \cdot \left(\frac{N}{N-\ell}\right)^{\hat{n}} \cdot \binom{N-\hat{n}}{\ell}}{\binom{128\sqrt{n}+1}{32\sqrt{n}} \cdot \binom{N}{\ell + 32\sqrt{n} \cdot \frac{\sqrt{n}}{64}}} \\ &= \frac{n^{\sqrt{n}} \cdot \left(\frac{N}{N-\ell}\right)^{\hat{n}}}{O(n^{\sqrt{n}/8}) \cdot n^{o(\sqrt{n})} \cdot 2^{O(\sqrt{n})}} \cdot \frac{\binom{N-\hat{n}}{\ell}}{\binom{N}{\ell+0.5n}} \quad \text{since } \binom{128\sqrt{n}+1}{32\sqrt{n}} = 2^{\Theta(\sqrt{n})} \\ &= \frac{\left(\frac{N}{\ell}\right)^{\hat{n}}}{O(n^{\sqrt{n}/8}) \cdot n^{o(\sqrt{n})} \cdot 2^{O(\sqrt{n})}} \cdot \frac{(N-\hat{n})!}{N!} \cdot \frac{(N-\ell-0.5n)!}{(N-\ell-\hat{n})} \cdot \frac{(\ell+0.5n)!}{\ell!} \quad \text{using (9)} \\ &\approx \frac{1}{O(n^{\sqrt{n}/8}) \cdot n^{o(\sqrt{n})} \cdot 2^{O(\sqrt{n})}} \cdot \left(\frac{N}{\ell}\right)^{\hat{n}} \cdot \frac{1}{N^{\hat{n}}} \cdot (N-\ell)^{\hat{n}-0.5n} \cdot \ell^{0.5n} \quad \text{by Thm 4.1} \\ &= \frac{1}{O(n^{\sqrt{n}/8}) \cdot n^{o(\sqrt{n})} \cdot 2^{O(\sqrt{n})}} \cdot (N-\ell)^{\hat{n}-0.5n} \cdot \ell^{0.5n-\hat{n}} \end{aligned}$$

$$\begin{aligned}
 &= \frac{1}{O(n^{\sqrt{n}/8}) \cdot n^{o(\sqrt{n})} \cdot 2^{O(\sqrt{n})}} \cdot \left(\frac{N - \ell}{\ell}\right)^{\hat{n} - 0.5n} \\
 &= \frac{1}{O(n^{\sqrt{n}/8}) \cdot n^{o(\sqrt{n})} \cdot 2^{O(\sqrt{n})}} \cdot \left(\frac{N - \frac{N}{2} \left(1 - \frac{\ln n}{\Gamma\sqrt{n}}\right)}{\frac{N}{2} \left(1 - \frac{\ln n}{\Gamma\sqrt{n}}\right)}\right)^{\hat{n} - 0.5n} && \text{using (8)} \\
 &= \frac{1}{O(n^{\sqrt{n}/8}) \cdot n^{o(\sqrt{n})} \cdot 2^{O(\sqrt{n})}} \cdot \left(\frac{1 + \frac{\ln n}{\Gamma\sqrt{n}}}{1 - \frac{\ln n}{\Gamma\sqrt{n}}}\right)^{\hat{n} - 0.5n} \\
 &\approx \frac{1}{O(n^{\sqrt{n}/8}) \cdot n^{o(\sqrt{n})} \cdot 2^{O(\sqrt{n})}} \cdot e^{2 \cdot \frac{\ln n}{\Gamma\sqrt{n}} \cdot (\hat{n} - 0.5n)} \\
 &= \frac{1}{O(n^{\sqrt{n}/8}) \cdot n^{o(\sqrt{n})} \cdot 2^{O(\sqrt{n})}} \cdot n^{\frac{2}{\Gamma\sqrt{n}}(n - 64\sqrt{n} - 0.5n)} && \text{using (7)} \\
 &= \frac{1}{O(n^{\sqrt{n}/8}) \cdot n^{o(\sqrt{n})} \cdot 2^{O(\sqrt{n})}} \cdot n^{\frac{2}{\Gamma}(\sqrt{n} - 64 - 0.5\sqrt{n})}.
 \end{aligned}$$

Using the estimate for  $\Gamma$  from (9), we obtain  $S \geq n^{\Omega(\sqrt{n})} = N^{\Omega(\sqrt{n})}$ , as desired. ◀

### 4.1 Remarks and Discussion

Notably, our lower bound only applies to circuits with bottom fan-in below  $\sqrt{N}$  – or rather, at most  $N^{1/2 - \Theta(1)}$ . This is a somewhat strong restriction because in a general depth-five circuit on  $N$  variables this fan-in could have been as high as  $N$ . In particular it is a stronger restriction than in the Kayal–Saha lower bound for certain Nisan–Wigderson polynomials (NW polynomials) [12], where this bottom fan-in had to be at most  $N^{1 - \Theta(1)}$ .

On the positive side, our lower bound works for arithmetic circuits over an arbitrary field, whereas the Kayal–Saha bound requires characteristic zero. Ultimately, this is because the technique for lower-bounding DPSP that they use (which is borrowed from Kayal *et al.* [10]) hinges on an operator-theoretic interpretation of matrix rank. In contrast, the DPSP lower bound that we use (borrowed from Kumar and Saraf [16]) is proven using counting alone.

It is worth understanding why our result hits a barrier at bottom fan-in around  $N^{1/2}$ . The random restriction used in this analysis retains at least one variable from almost every row of every matrix in the IMM polynomial. Therefore, it reduces the variable set from size  $N$  to size slightly more than  $N^{1/2}$  (the “slightly” is in fact contingent on making  $q$  very large), and this is not a severe enough random restriction to give us the required circuit decomposition. More concretely, satisfying Equation (21), even in the extreme setting  $b = 0$ , forces  $q \rightarrow \infty$  as  $\mu \rightarrow 1/2$ . It could be that an even more severe random restriction is worth considering, but proving a good DPSP lower bound for IMM polynomials so restricted seems unlikely to proceed along the lines of the Kumar–Saraf argument. Whether our size lower bound still holds with the bottom fan-in allowed to reach up to  $N^{1 - \Theta(1)}$ , or even  $N$  (which is the general case) is the most immediate and natural open question.

**Acknowledgments.** The second author would like to thank Neeraj Kayal, Chandan Saha, and Ramprasad Saptharishi for several stimulating discussions.

---

### References

- 1 Manindra Agrawal and V Vinay. Arithmetic circuits: A chasm at depth four. In *Proc. 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 67–75, 2008.

- 2 Stuart J. Berkowitz. On computing the determinant in small parallel time using a small number of processors. *Inform. Process. Lett.*, 18(3):147–150, 1984.
- 3 Devdatt P Dubhashi and Alessandro Panconesi. *Concentration of measure for the analysis of randomized algorithms*. Cambridge University Press, 2009.
- 4 Hervé Fournier, Nutan Limaye, Guillaume Malod, and Srikanth Srinivasan. Lower bounds for depth 4 formulas computing iterated matrix multiplication. In *Proc. 46th Annual ACM Symposium on the Theory of Computing*, pages 128–135, 2014.
- 5 Dima Grigoriev and Marek Karpinski. An exponential lower bound for depth 3 arithmetic circuits. In *Proc. 30th Annual ACM Symposium on the Theory of Computing*, pages 577–582, 1998.
- 6 Dima Grigoriev and Alexander Razborov. Exponential lower bounds for depth 3 arithmetic circuits in algebras of functions over finite fields. *Applicable Algebra in Engineering, Communication and Computing*, 10(6):465–487, 2000.
- 7 Ankit Gupta, Pritish Kamath, Neeraj Kayal, and Ramprasad Saptharishi. Approaching the chasm at depth four. In *Proc. 28th Annual IEEE Conference on Computational Complexity*, pages 65–73, 2013.
- 8 Ankit Gupta, Pritish Kamath, Neeraj Kayal, and Ramprasad Saptharishi. Arithmetic circuits: A chasm at depth three. In *Proc. 54th Annual IEEE Symposium on Foundations of Computer Science*, pages 578–587, 2013.
- 9 Kumar Joag-Dev and Frank Proschan. Negative association of random variables, with applications. *Ann. Stat.*, 11(1):286–295, 1983.
- 10 Neeraj Kayal, Nutan Limaye, Chandan Saha, and Srikanth Srinivasan. An exponential lower bound for homogeneous depth four arithmetic formulas. In *Proc. 55th Annual IEEE Symposium on Foundations of Computer Science*, 2014.
- 11 Neeraj Kayal, Nutan Limaye, Chandan Saha, and Srikanth Srinivasan. Super-polynomial lower bounds for depth-4 homogeneous arithmetic formulas. In *Proc. 46th Annual ACM Symposium on the Theory of Computing*, pages 119–127, 2014.
- 12 Neeraj Kayal and Chandan Saha. Lower bounds for depth three arithmetic circuits with small bottom fanin. Technical Report TR14-089, ECCC, 2014.
- 13 Neeraj Kayal, Chandan Saha, and Ramprasad Saptharishi. A super-polynomial lower bound for regular arithmetic formulas. In *Proc. 46th Annual ACM Symposium on the Theory of Computing*, pages 146–153, 2014.
- 14 Neeraj Kayal and Ramprasad Saptharishi. *A selection of lower bounds for arithmetic circuits*, pages 77–115. Springer Verlag, April 2014.
- 15 Pascal Koiran. Arithmetic circuits: The chasm at depth four gets wider. *Theor. Comput. Sci.*, 448:56–65, 2012.
- 16 Mrinal Kumar and Shubhangi Saraf. On the power of homogeneous depth 4 arithmetic circuits. In *Proc. 55th Annual IEEE Symposium on Foundations of Computer Science*, 2014.
- 17 Noam Nisan and Avi Wigderson. Hardness vs. randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994.
- 18 Noam Nisan and Avi Wigderson. Lower bounds on arithmetic circuits via partial derivatives. In *Proc. 36th Annual IEEE Symposium on Foundations of Computer Science*, pages 16–25, 1995.
- 19 Ramprasad Saptharishi. Recent progress on arithmetic circuit lower bounds. *Bulletin of the EATCS*, 114, 2014.
- 20 Amir Shpilka and Amir Yehudayoff. Arithmetic circuits: A survey of recent results and open questions. *Found. Trends Theor. Comput. Sci.*, 5(3-4):207–388, 2010.

- 21 Sébastien Tavenas. Improved bounds for reduction to depth 4 and depth 3. In *Proc. 38th International Symposium on Mathematical Foundations of Computer Science*, pages 813–824, 2013.
- 22 L. G. Valiant. Completeness classes in algebra. In *Proc. 11th Annual ACM Symposium on the Theory of Computing*, pages 249–261, 1979.
- 23 Leslie G. Valiant, Sven Skyum, S. Berkowitz, and Charles Rackoff. Fast parallel computation of polynomials using few processors. *SIAM J. Comput.*, 12(4):641–644, 1983.

# Factors of Low Individual Degree Polynomials\*

Rafael Oliveira

Department of Computer Science, Princeton University  
35 Olden St., Princeton NJ, USA  
rmo@cs.princeton.edu

---

## Abstract

In [8], Kaltofen proved the remarkable fact that multivariate polynomial factorization can be done efficiently, in randomized polynomial time. Still, more than twenty years after Kaltofen's work, many questions remain unanswered regarding the complexity aspects of polynomial factorization, such as the question of whether factors of polynomials efficiently computed by arithmetic formulas also have small arithmetic formulas, asked in [10], and the question of bounding the depth of the circuits computing the factors of a polynomial.

We are able to answer these questions in the affirmative for the interesting class of polynomials of bounded individual degrees, which contains polynomials such as the determinant and the permanent. We show that if  $P(x_1, \dots, x_n)$  is a polynomial with individual degrees bounded by  $r$  that can be computed by a formula of size  $s$  and depth  $d$ , then any factor  $f(x_1, \dots, x_n)$  of  $P(x_1, \dots, x_n)$  can be computed by a formula of size  $\text{poly}((rn)^r, s)$  and depth  $d+5$ . This partially answers the question above posed in [10], that asked if this result holds without the exponential dependence on  $r$ . Our work generalizes the main factorization theorem from Dvir et al. [2], who proved it for the special case when the factors are of the form  $f(x_1, \dots, x_n) \equiv x_n - g(x_1, \dots, x_{n-1})$ . Along the way, we introduce several new technical ideas that could be of independent interest when studying arithmetic circuits (or formulas).

**1998 ACM Subject Classification** F.1.1 Models of Computation, F.1.3 Complexity Measures and Classes, F.2.1 Numerical Algorithms and Problems

**Keywords and phrases** Arithmetic Circuits, Factoring, Algebraic Complexity

**Digital Object Identifier** 10.4230/LIPIcs.CCC.2015.198

## 1 Introduction

Let  $f(x_1, \dots, x_n) \in \mathbb{F}[x_1, \dots, x_n]$  be a multivariate polynomial over a field  $\mathbb{F}$ . The individual degree of  $f$  with respect to variable  $x_i$ , denoted by  $\deg_{x_i}(f)$ , is the largest power of  $x_i$  appearing in a monomial of  $f$ . Many interesting polynomials have bounded individual degree, such as the Permanent and Determinant polynomials. Moreover, the class of polynomials of bounded individual degree is closed under factorization, since if a polynomial  $f(x_1, \dots, x_n)$  has individual degrees bounded by  $r$ , so will its factors. In this work, we study the problem of formula (circuit) factorization of polynomials of low individual degree.

One of the basic operations on polynomials is factorization. This problem can be phrased as follows: given a polynomial  $P(x_1, \dots, x_n)$ , decide whether  $P(x_1, \dots, x_n)$  is irreducible, or if not, output one of its factors, which we denote by  $f(x_1, \dots, x_n)$ . From the computational perspective, we will usually be given a device computing the polynomial  $P$ , and we will be asked to output a similar device computing  $f$ . In the field of arithmetic complexity, the most natural device for computing polynomials is an arithmetic circuit or a formula (see

---

\* Research supported by NSF grant CCF-1217416 and by the Sloan fellowship.





(Definition 1.1 below). Therefore, we will assume that we are given  $P$  as an arithmetic circuit (formula) and output one of its factors in the same representation. We now give the definition of an arithmetic circuit/formula:

► **Definition 1.1.** An *arithmetic circuit*  $\Gamma$  is a directed acyclic labeled graph in which the vertices are called ‘gates’. The gates of  $\Gamma$  with in-degree 0 are called *inputs* and are labeled by either a variable from  $\{x_1, \dots, x_n\}$  or by a field element from  $\mathbb{F}$ . Every other gate of  $\Gamma$  is labeled by either ‘ $\times$ ’ or ‘ $+$ ’ and has in-degree 2. (If we talk about bounded depth circuits/formulas, then we remove the restriction on the in-degree.) There is one gate with out-degree 0, which we call the *output gate*. Each gate in  $\Gamma$  computes a polynomial in  $\mathbb{F}[x_1, \dots, x_n]$  in the natural way. An arithmetic circuit is called a *formula* if its underlying graph is a tree. The *size* of a circuit (formula)  $\Gamma$ , written  $|\Gamma|$ , is given by the number of edges in the circuit (formula) and the *depth* of  $\Gamma$ , written  $\text{depth}(\Gamma)$ , is defined as the length of the longest directed path in the graph of  $\Gamma$ .

Polynomial factorization is one of the cornerstone problems in modern computer algebra, and as such has been the focus of intensive research. The past three decades have seen major advances on the development of efficient algorithms for polynomial factorization, pioneered by the works of Lenstra et al. and Kaltofen [11, 7, 8, 9]. In addition to the general problem, polynomial factorization has also been studied in many other important (and more restricted) representations. For instance, in the sparse representation, where the input polynomial is given as a list of its coefficients and monomials, the works of Lenstra, Kaltofen and von zur Gathen [12, 4] give efficient algorithms for sparse factorization in the univariate and in the multivariate cases. For a more complete survey on polynomial factorization we refer the reader to the survey [9] and to the book [3].

In the seminal work of Kaltofen [8], it is proved that if  $P(x_1, \dots, x_n)$  of total degree  $D$  can be computed by an arithmetic circuit of size  $s$ , then any of its factors have arithmetic circuits of size  $\text{poly}(n, s, D)$ . Moreover, Kaltofen gives a randomized algorithm that with high probability outputs such a factor in polynomial time. This result, besides settling an important complexity theoretic question, has since then had a great impact in many areas of computer science, such as coding theory [16, 5], derandomization [6] and cryptography [1]. However, many interesting questions on the complexity of arithmetic circuits or formulas under factorization remain unanswered. In particular, we study the following two questions, where the first one was asked in the work of Kopparty et al. [10], while the second question was stated as an open problem in the survey [15, Open Problem 19]:

1. If  $P(x_1, \dots, x_n)$  of total degree  $D$  is computed by an arithmetic formula of size  $s$ , is it true that any of its factors will also have formulas of size  $\text{poly}(n, s, D)$ ?
2. If  $P(x_1, \dots, x_n)$  can be computed by a circuit of size  $s$  and depth  $d$ , can its factors be computed by a circuit of size  $\text{poly}(s)$  and depth  $O(d)$ ?

In this work, we answer both of these questions in the affirmative, in the case where the input polynomial  $P$  has bounded individual degrees. In particular, we show:

► **Theorem 1.2.** *Let  $P(x_1, \dots, x_n) \in \mathbb{F}[x_1, \dots, x_n] \setminus \{0\}$  be such that  $\deg_{x_i}(P) \leq r$ ,  $1 \leq i \leq n$ , and let  $f(x_1, \dots, x_n) \in \mathbb{F}[x_1, \dots, x_n]$  be a factor of  $P$ , where  $\mathbb{F}$  is a field of characteristic zero. If there exists a formula (circuit) of size  $s$  and depth  $d$  computing  $P$ , then there exists a formula (circuit) of depth  $d+5$  and size  $\text{poly}((nr)^r, s)$  that computes  $f(x_1, \dots, x_n)$ . Moreover, if we require the in-degree of each gate to be 2, then the size remains the same and the depth becomes  $d + O(r \log(nr))$ .*

Notice that our theorem has no restriction on the individual degrees of the polynomials computed by the intermediate gates of the circuit (that is, we have no syntactic restrictions). We only care about the individual degrees of the output polynomial, which we regard as bounded by a constant, denoted by  $r$ , in the theorem above.

Theorem 1.2 provides a direct answer to the second question posed above in the case where  $P$  has bounded individual degrees (that is,  $r$  is a constant). The connection between Theorem 1.2 and the first question comes from the fact that one can always balance formulas to have logarithmic depth. More precisely, suppose that we are given a formula  $\Phi$  (with in-degree bounded by 2) of size  $s = \text{poly}(n)$  computing  $P$ . By Theorem 2.7 in [15], we can assume that  $\Phi$  is of size  $\text{poly}(s)$  and  $\text{depth}(\Phi) = O(\log s)$ . Hence, Theorem 1.2 implies that there exists a formula  $\Psi$ , with in-degree bounded by 2, of depth  $\text{depth}(\Psi) = \text{depth}(\Phi) + O(r \log(sn)) = O(\log s)$  and size  $\text{poly}((nr)^r, s) = \text{poly}(s)$  computing any factor  $f(x_1, \dots, x_n)$  of  $P$ . This provides an affirmative answer to the first question.

Before giving an overview of the proof of Theorem 1.2, we give some background on related work on factorization in general and in bounded depth circuits.

The problem of factoring in bounded depth was studied previously in [2], who showed that if  $P(x_1, \dots, x_n)$  has a depth  $d$  circuit of size  $s$  and  $\deg_{x_n}(P) \leq r$ , then its factors of the form  $x_n - \phi(x_1, \dots, x_{n-1})$  have depth  $d + 3$  circuits of size  $\text{poly}(n^r, s)$ . This result was used to extend the hardness-randomness tradeoffs of [6] to the bounded depth model. Our main theorem generalizes their result to any factor of  $P$ , provided that  $P$  has bounded individual degrees.

Shpilka and Volkovich in [14] initiated the study of factorization of multilinear polynomials, which are the most basic case of polynomials of bounded individual degrees. They relate the problem of deterministically factoring multilinear polynomials to the problem of performing deterministic Polynomial Identity Testing (PIT). In their paper, they prove that these two problems are roughly equivalent in the multilinear setting for most restricted multilinear circuit classes that have been studied. Since the problem of performing deterministic PIT seems to be hard, even for the class of multilinear formulas, this shed some light on the difficulty of obtaining deterministic factorization even for this model. This equivalence between deterministic PIT and deterministic polynomial factorization was later generalized by Kopparty et al. in [10] to polynomials (of polynomial degree) computed by general circuits. Since we prove here that, for polynomials of bounded individual degrees computed by circuits of small depth, their factors can also be computed by circuits of small depth, one could hope for similar connections between PIT for restricted classes of circuits – say of bounded depth and low individual degrees – and factorization of polynomials in such classes.

## 2 Proof Overview

In this section, we give an overview of the proof of the main theorem. For simplicity of exposition, we will only refer to arithmetic circuits in this overview, but our results hold true for formulas as well, as the statements in the later sections show. We begin with a definition:

► **Definition 2.1** (Approximate Root). Let  $P(x_1, \dots, x_n, y)$  be a polynomial in  $\mathbb{F}[x_1, \dots, x_n, y]$ . We say that  $q(x_1, \dots, x_n)$  is a *root of  $P$  up to degree  $t$*  if all the homogeneous parts up to degree  $t$  of the polynomial  $P(x_1, \dots, x_n, q(x_1, \dots, x_n))$  are zero. That is,  $P(x_1, \dots, x_n, q(x_1, \dots, x_n))$  only has monomials of degree larger than  $t$ .

Given a polynomial  $P(x_1, \dots, x_n, y) \in \mathbb{F}[x_1, \dots, x_n, y]$  with individual degree in  $y$  bounded by  $r$ , Dvir et al. [2] show that if  $P(0, \dots, 0, y)$  has no double roots, that is,  $P(0, \dots, 0, y)$  can

be factored as

$$P(0, \dots, 0, y) \equiv c \cdot \prod_{i=1}^r (y - \mu_i)$$

where  $\mu_i \neq \mu_j$  for  $i \neq j$ , then for each  $\mu_i$ , there exists an approximate root  $q_{i,t}(x_1, \dots, x_n)$  of  $P$  up to degree  $t$  such that  $q_{i,t}(0, \dots, 0) = \mu_i$ . Moreover, they show that if  $P$  is computed by a circuit  $\Gamma$  of size  $s$  and depth  $d$ , then there exists a circuit of size  $\text{poly}(t^r, s)$  and depth  $d + 2$  computing  $q_{i,t}(x_1, \dots, x_n)$ .

With this idea in mind, suppose for simplicity that

$$P(x_1, \dots, x_n, y) \equiv \prod_{i=1}^r (y - g_i(x_1, \dots, x_n)),$$

where each polynomial  $g_i(x_1, \dots, x_n)$  has a nonzero constant term  $\mu_i$  and  $\mu_i \neq \mu_j$  for  $i \neq j$ . In this case we are in the framework of [2], since

$$P(0, \dots, 0, y) \equiv \prod_{i=1}^r (y - \mu_i)$$

and the roots  $\mu_i$  are distinct. As Section 4 shows, we can guarantee distinct roots in  $P(0, \dots, 0, y)$  by using a random shift of the variables  $(x_1, \dots, x_n)$ , as long as  $P$  is square-free. Therefore, for each  $\mu_i$  and  $t \geq 1$ , we can find polynomials  $q_{i,t}(x_1, \dots, x_n)$  such that  $q_{i,t}(0, \dots, 0) = \mu_i$  and the polynomial  $P(x_1, \dots, x_n, q_{i,t}(x_1, \dots, x_n))$  only has terms of degree larger than  $t$ . Since

$$P(x_1, \dots, x_n, q_{i,t}(x_1, \dots, x_n)) \equiv \prod_{j=1}^r (q_{i,t}(x_1, \dots, x_n) - g_j(x_1, \dots, x_n)),$$

the minimum degree terms of  $P(x_1, \dots, x_n, q_{i,t}(x_1, \dots, x_n))$  must come from the product of the minimum degree terms of each of the polynomials  $q_{i,t}(x_1, \dots, x_n) - g_j(x_1, \dots, x_n)$ . Notice that for each  $j \neq i$ , the constant term of  $q_{i,t}(x_1, \dots, x_n) - g_j(x_1, \dots, x_n)$  is equal to  $\mu_i - \mu_j$ , which is nonzero. Therefore, the minimum degree terms of  $P(x_1, \dots, x_n, q_{i,t}(x_1, \dots, x_n))$  must come from the minimum degree terms of the polynomial  $q_{i,t}(x_1, \dots, x_n) - g_i(x_1, \dots, x_n)$ . Because  $P(x_1, \dots, x_n, q_{i,t}(x_1, \dots, x_n))$  only has terms of degree larger than  $t$ , the same must happen to the polynomial  $q_{i,t}(x_1, \dots, x_n) - g_i(x_1, \dots, x_n)$ . This implies that  $q_{i,t}(x_1, \dots, x_n)$  approximates the actual root  $g_i(x_1, \dots, x_n)$  of  $P$  up to degree  $t$ . Hence, if we pick  $t$  larger than the total degree of  $g_i$ , the lower degree terms of  $q_{i,t}$  correspond to the root  $g_i$ , and therefore we can recover this root  $g_i$  (and use it to factor  $P$ ).

There are two main issues with this approach that we need to overcome, if we are to generalize it. The first issue is that  $P$  may not factor into linear factors in  $y$ , that is, polynomials of the form  $y - g_i(x_1, \dots, x_n)$ . The second one is that  $P$  need not be monic in  $y$ , in which case we will still need to recover its leading coefficient – which is a polynomial in  $\mathbb{F}[x_1, \dots, x_n]$ .

To deal with the first issue, let us study a toy example: assume that  $P$  is monic in  $y$  with  $\deg_y(P) = r$ , that is,

$$P(x_1, \dots, x_n, y) \equiv y^r + \sum_{i=0}^{r-1} P_i(x_1, \dots, x_n)y^i,$$

but  $P$  does not factor into linear factors in  $y$ . Let  $f(x_1, \dots, x_n, y)$  be one of its factors, of degree  $k$  in  $y$ . Since  $P$  is monic in  $y$ , we know that  $f$  must also be monic in  $y$ . Note that

if we work over the algebraic closure of  $\mathbb{F}(x_1, \dots, x_n)$  (that is, the field  $\overline{\mathbb{F}(x_1, \dots, x_n)}$ ), we can factor  $P$  (and  $f$ ) into linear factors in  $y$ . In this work, we will not describe what the algebraic closure of  $\mathbb{F}[x_1, \dots, x_n]$  is, since it is a very complex field, and it is not needed in our proof. We only mention  $\overline{\mathbb{F}(x_1, \dots, x_n)}$  here to give us some intuition on how to generalize the root finding approach described above. For simplicity, think of elements of the closure as “functions” over the variables  $x_1, \dots, x_n$ . Since  $f$  divides  $P$ , if

$$P(x_1, \dots, x_n, y) \equiv \prod_{i=1}^r (y - \varphi_i(x_1, \dots, x_n)),$$

then there will be indices (say  $i$  from 1 to  $k$ ) such that

$$f(x_1, \dots, x_n, y) \equiv \prod_{i=1}^k (y - \varphi_i(x_1, \dots, x_n)).$$

However, it is worth noting that these linear factors will not be polynomials! Nevertheless, the fact that they share some roots in the closure of  $\mathbb{F}[x_1, \dots, x_n]$  gives us a hint on what to do next. To overcome this problem, we will (in Lemma 6.1 and Corollary 6.2) approximate these functions  $\varphi_i$  by polynomials  $g_{i,t}$ , in a way that the polynomial

$$g_t(x_1, \dots, x_n, y) \equiv \prod_{i=1}^k (y - g_{i,t}(x_1, \dots, x_n))$$

agrees with  $f$  on the terms of order smaller than  $t$ . Therefore, for large enough  $t$ , the lower order terms of  $g_t(x_1, \dots, x_n, y)$  will correspond to the polynomial  $f$ , which we can then obtain by interpolation (Lemma 3.3). We can think of each polynomial  $g_{i,t}$  as the Taylor expansion of  $\varphi_i$  up to degree  $t$ .

The way we obtain these approximations to the roots (the polynomials  $g_{i,t}$ ) is by a procedure similar in nature to Hensel lifting. Suppose that  $\varphi_i(0, \dots, 0) = \mu_i$  for  $1 \leq i \leq k$ , and moreover, suppose that  $\mu_i \neq \mu_j$  for  $i \neq j$ . From each valuation  $\mu_i$ , we will construct a family of polynomials  $g_{i,t}$  of degree  $t$ , such that  $g_{i,t}(x_1, \dots, x_n)$  is a root of  $f$  up to degree  $t$ . Now, the question is: how can we construct this family of polynomials if we do not have access to  $f$ ? The answer to this question lies on the fact that each root  $y - \varphi_i$  of  $f$  is also a root of  $P$ , and therefore we can access the valuations of  $\varphi_i$ 's through the circuit computing  $P$ . Hence, we will use the fact that the  $\varphi_i$ 's are also roots of  $P$  in order to find the polynomials  $g_t$  that approximate  $f$  (Lemma 7.1).

To overcome the second main issue, that the polynomial  $P$  may not be monic, let us define

$$f(x_1, \dots, x_n, y) \equiv \sum_{i=0}^k f_i(x_1, \dots, x_n) y^i \quad \text{and} \quad P(x_1, \dots, x_n, y) \equiv \sum_{i=0}^r P_i(x_1, \dots, x_n) y^i,$$

where  $f_k(x_1, \dots, x_n) \not\equiv 0$  and  $P_r(x_1, \dots, x_n) \not\equiv 0$ . If  $f$  divides  $P$ , then it must be the case that the leading coefficient  $f_k$  of  $f$  divides the leading coefficient  $P_r$  of  $P$ . Hence, a possible solution to this second issue would be to find, by some kind of induction, a small circuit for  $f_k$  based on the circuit for  $P_r$  that we obtain from  $P$ . Then, we could generalize the factoring result for monic polynomials to the case where the factors are rational functions of the form

$$\frac{f(x_1, \dots, x_n, y)}{f_k(x_1, \dots, x_n)} \equiv y^k + \sum_{i=0}^{k-1} \frac{f_i(x_1, \dots, x_n)}{f_k(x_1, \dots, x_n)} y^i.$$

With these two results, we could multiply the circuits computing  $f_k$  and  $\frac{f}{f_k}$  to obtain our factor  $f$ .

More precisely, if we could find, by induction on the number of variables, a small circuit  $\Phi_k$  for  $f_k$  based on the circuit  $\Gamma_r$  for  $P_r$  that we obtain from  $P$  via interpolation (Lemma 3.4), and if we could find a small circuit  $\Upsilon$  for the rational function  $\frac{f}{f_k}$  based on the circuit  $\Gamma$  computing  $P$  (Lemma 7.1), then the circuit given by  $\Upsilon \times \Phi_k$  would compute the polynomial  $f$ , as we wanted.

One problem with this approach is that, even if we can generalize the monic factoring result to monic rational functions as above, as far as we know, the best bound on the size of the circuit  $\Gamma_r$  computing  $P_r$  is given by  $3r \cdot s$  (see Lemma 3.4). Therefore, if we define  $T(n, s)$  as the maximum size of a factor of a polynomial in  $n$  variables computed by a circuit of size  $s$ , the induction given by the procedure above would give us the following bounds on the size:

$$T(n + 1, s) \leq T(n, 3r \cdot s) + \text{poly}((nr)^r, s).$$

The reason for this bound is the following:  $P(x_1, \dots, x_n, y)$  has  $n + 1$  variables and is computed by  $\Gamma$ , which has size  $s$ . Hence, the maximum size of a factor  $f$  is by definition  $T(n + 1, s)$ . Since  $f_k$  divides the leading coefficient  $P_r$ , which is computed by  $\Gamma_r$  of size  $3rs$  and has  $n$  variables, the bound we have on the size of  $\Phi_k$  is given by  $T(n, 3rs)$ , because now the input polynomial is  $P_r$ . Assuming that the size of  $f/f_k$  can be bounded by  $((nr)^r \cdot s)^\alpha$ , for some constant  $\alpha$  (which we can by Lemma 7.1), we obtain the additive factor  $\text{poly}((nr)^r, s)$ . Since the circuit for  $f$  is given by  $\Upsilon \times \Phi_k$ , we need to add the bounds on the sizes for  $\Phi_k$  and  $\Upsilon$ . However, when we solve this equation, we obtain that

$$T(n + 1, s) \leq T(1, (3r)^n \cdot s) + \text{poly}((nr)^r, (3r)^n \cdot s)$$

which is exponential in  $n$ , the number of variables! Therefore, this approach, as it is, cannot work.

The main problem with the recursion above is that the bound on the circuit size of the leading coefficient, if we only use Lemma 3.4, keeps getting worse as we reduce the number of variables – it will become  $(3r)^\ell \cdot s$  if we get rid of  $\ell$  variables. To get around this issue, we define the *reversal* of a polynomial with respect to a specific variable and we study its properties with regards to divisibility. If

$$P(x_1, \dots, x_n, y) \equiv \sum_{i=0}^r P_i(x_1, \dots, x_n) y^i$$

is a polynomial, with  $P_r(x_1, \dots, x_n) \cdot P_0(x_1, \dots, x_n) \neq 0$ , we define its reversal with respect to  $y$  as the polynomial

$$\tilde{P}(x_1, \dots, x_n, y) \equiv \sum_{i=0}^r P_i(x_1, \dots, x_n) y^{r-i}.$$

That is,  $\tilde{P}$  is obtained from the polynomial  $P$  by “reversing” the coefficients  $P_i(x_1, \dots, x_n)$ . It is easy to see that  $f$  divides  $P$  iff  $\tilde{f}$  divides  $\tilde{P}$ . By performing a reversal, notice that we have transformed the leading coefficient of our problem from  $P_r(x_1, \dots, x_n)$  to  $P_0(x_1, \dots, x_n)$ . This has the advantage that now, the leading coefficient of our input polynomial can be computed by the circuit  $\Gamma|_{y=0}$  (that is, the circuit obtained from  $\Gamma$  by setting  $y = 0$ ), which has size  $\leq s$ . This now allows us to recurse into the division of  $f_0$  by  $P_0$  (the new leading

coefficients after the reversal) without paying the multiplicative cost on the size of the circuit. Hence with this idea we avoid paying the exponential blowup on the circuit size! On the coin side, notice that the size of the circuit computing the polynomial  $\tilde{P}$  is bounded by  $8r^2 \cdot s$ , according to Lemma 3.7. But this blow up does not hurt us, since the reversal is not cumulative.

More precisely, we now have the following recursion: we want to bound the size of a factor of  $P$ , computed by a circuit  $\Gamma$  of size  $s$  and on  $n + 1$  variables. This bound is by definition  $T(n + 1, s)$ . Let  $\tilde{\Gamma}$  be a circuit computing  $\tilde{P}$ . Suppose we can find a circuit computing  $f/f_0$  of size bounded by  $((nr)^r \cdot |\tilde{\Gamma}|)^\alpha \leq ((nr)^r \cdot 8r^2s)^\alpha$ , for some constant  $\alpha$  (which we can by Lemma 7.1). Then we are only left with the problem of finding a small circuit for  $f_0$ , which divides  $P_0$ , which in turn can be computed by a circuit of size bounded by  $s$  in  $n$  variables. The bound for a circuit for  $f_0$  is given in this case by  $T(n, s)$ , by definition of the function  $T$ . Therefore, our recursion becomes

$$T(n + 1, s) \leq T(n, s) + ((nr)^r \cdot 8r^2 \cdot s)^\alpha$$

which implies that

$$T(n, s) \leq n \cdot ((nr)^r \cdot 8r^2 \cdot s)^\alpha = \text{poly}((nr)^r, s),$$

as we wanted!

The idea of the reversal of a polynomial is similar to the definition of *reversal* of a univariate polynomial given in [3, §9.1]. This notion of reversal is used there to perform division with remainder for univariate polynomials by using Newton iteration.

To generalize the monic factoring result to the case when  $f$  is monic in  $y$  with rational coefficients, we introduce the idea of an approximation polynomial of a rational function (see Section 5), and we use this approximation polynomial in Lemma 7.1 (instead of the rational function) as the “factor” of the input polynomial. If  $f$  is a rational function of the form

$$f(x_1, \dots, x_n, y) \equiv \frac{1}{1 - g(x_1, \dots, x_n)} \cdot \sum_{i=0}^k f_i(x_1, \dots, x_n)y^i,$$

where  $g(x_1, \dots, x_n)$  and  $f_i(x_1, \dots, x_n)$  are polynomials in  $\mathbb{F}[x_1, \dots, x_n]$  such that  $g(0, \dots, 0) = 0$ , we define its approximation polynomial (to degree  $m$ ) as the following polynomial

$$\psi_{f,m}(x_1, \dots, x_n, y) \equiv (1 + g + g^2 + \dots + g^m) \cdot \sum_{i=0}^k f_i y^i,$$

where  $g \equiv g(x_1, \dots, x_n)$  and  $f_i \equiv f_i(x_1, \dots, x_n)$ . This polynomial “approximates” the rational function  $f(x_1, \dots, x_n, y)$  in the sense that, for large enough  $m$ , the polynomial obtained by  $\psi_{f,m}(x_1, \dots, x_n, y) \cdot (1 - g(x_1, \dots, x_n))$  is equal to  $f(x_1, \dots, x_n) \cdot (1 - g(x_1, \dots, x_n))$ , up to high order terms (see Observation 5.3), which we can get rid of by interpolation (Lemma 3.3). By adapting the approach in [2] to work with approximation polynomials, we can find all the “roots” of the approximation polynomials, and after that combine this approximation polynomial with the circuit obtained to compute the leading term.

After we take care of finding the leading coefficient  $f_0(x_1, \dots, x_n)$  (of the reversed polynomial  $\tilde{f}(x_1, \dots, x_n, y)$ ), and after recovering the approximation polynomial  $\psi_{\tilde{f},m}$  (see Lemma 7.1), we can multiply it by  $f_0$  to obtain the factor  $f$  (up to high order terms) which, after interpolation, becomes our desired factor (see Theorem 7.2).

We conclude this proof outline with a basic roadmap of the main ideas involved in this work:

1. Given a circuit  $\Gamma$  for our polynomial  $P(x_1, \dots, x_n, y)$ , we find a circuit  $\tilde{\Gamma}$  computing the reversal polynomial  $\tilde{P}(x_1, \dots, x_n, y)$ . (Lemma 3.7)
2. We use the circuit  $\tilde{\Gamma}$  to find small circuits  $\Phi_{i,t}$  for each approximate root of  $\tilde{P}$  up to degree  $t$ . (Section 6)
3. Since  $\tilde{f}$ , divides  $\tilde{P}$  (Lemma 3.8), any approximate root of  $\tilde{f}$  will also be an approximate root of  $\tilde{P}$ . By combining the circuits  $\Phi_{i,t}$  computing the approximate roots of  $\tilde{f}(x_1, \dots, x_n, y)$ , find circuit  $\Psi$  computing the approximation polynomial (see Section 5) of the monic rational function  $\frac{\tilde{f}(x_1, \dots, x_n, y)}{f_0(x_1, \dots, x_n)}$ . (Lemma 7.1)
4. By induction, obtain the circuit  $\Lambda_0$  computing  $f_0(x_1, \dots, x_n)$ , through the circuit  $\Gamma|_{y=0}$  computing  $P_0(x_1, \dots, x_n) \equiv P(x_1, \dots, x_n, 0)$ .
5. We then prove that the lower order terms of the circuit  $\Phi = \Lambda_0 \times \Psi$  compute the polynomial  $\tilde{f}$ . (Theorem 7.2)
6. By interpolation (Lemma 3.3) and by the Reversal Lemma (Lemma 3.7), obtain the lower order terms from  $\Phi$  computing  $f$ .

## 2.1 Organization

The rest of the paper is organized as follows: in Section 3, we set up notations, go over some useful background and discuss the concept of reversal of a polynomial. In Section 4, we introduce the concept of properly splitting variable restrictions. In Section 5, we formally introduce the concepts of standard forms and approximation polynomials. In Section 6, we adapt the approach of [2] to find small formulas for the roots of  $P(x_1, \dots, x_n, y)$ . In Section 7 we prove our main technical lemma and theorem. In Section 8, we conclude and propose some open problems.

For the sake of brevity of exposition, we only give a proof of our main technical theorem. The proofs of all other facts stated in this paper can be found in the full version [13].

## 3 Preliminaries

In this section, we establish the notation that will be used throughout the paper and some technical background that will be needed in the proof of our main theorem.

### 3.1 Notations

From this point on, we will use boldface for vectors, and regular font for scalars. Thus, we will denote the vector  $(x_1, \dots, x_n)$  by  $\mathbf{x}$ . If we want to multiply the vector  $\mathbf{x}$  by a scalar  $z$  we will denote this product by  $z\mathbf{x}$ .

We will denote our base field by  $\mathbb{F}$ , assume that  $\mathbb{F}$  has characteristic zero and that it is algebraically closed. The results in this paper also hold for non-closed fields of large enough characteristic, if we allow ourselves to use elements from field extensions. The assumptions just made are for clarity of exposition.

Let  $\mathbb{N}_0$  be the set of natural numbers including zero, that is,  $\mathbb{N}_0 = \{0, 1, 2, 3, \dots\}$ . If  $\mathbf{e} \in \mathbb{N}_0^n$  is a vector of natural numbers and  $\mathbf{x} = (x_1, \dots, x_n)$  is a vector of formal variables, we define  $\mathbf{x}^{\mathbf{e}} = \prod_{i=1}^n x_i^{e_i}$ . That is,  $\mathbf{x}^{\mathbf{e}}$  is the monomial corresponding to the product of the variables  $\prod_{i=1}^n x_i^{e_i}$ , where each variable is raised to the proper power.

We will denote  $\mathbb{F}(\mathbf{x})[y]$  as the set of polynomials in the variable  $y$  whose coefficients are rational functions on the variables  $\mathbf{x}$ . That is,  $f(\mathbf{x}, y) \in \mathbb{F}(\mathbf{x})[y]$  iff it can be expressed in the form  $f(\mathbf{x}, y) \equiv \sum_{i=0}^k \frac{f_i(\mathbf{x})}{g_i(\mathbf{x})} y^i$ , with  $f_i(\mathbf{x}), g_i(\mathbf{x}) \in \mathbb{F}[\mathbf{x}], 0 \leq i \leq k$ .

When working with a polynomial in  $\mathbb{F}[\mathbf{x}, y]$ , we might be interested in looking at the homogeneous parts of a polynomial with respect to certain variables only. This will be particularly useful when lifting the “roots” of a polynomial  $f(\mathbf{x}, y)$  of the form  $y - q(\mathbf{x})$  in order to obtain a circuit computing  $f(\mathbf{x}, y)$ . To this end, we introduce the following definition.

► **Definition 3.1** (Partial Homogeneous Parts). Let  $P(\mathbf{x}, y) \equiv \sum_{\mathbf{d}} \alpha_{\mathbf{d}}(y) \cdot \mathbf{x}^{\mathbf{d}}$  be a polynomial in  $\mathbb{F}[\mathbf{x}, y]$ , where each  $\alpha_{\mathbf{d}}(y) \in \mathbb{F}[y]$ . For each  $m \in \mathbb{N}_0$ , we define  $H_m^{\mathbf{x}}[P]$  as the polynomial formed by the homogeneous parts of degree  $m$  of  $P(\mathbf{x}, y)$ , when seen as a polynomial in  $\mathbb{F}[y][\mathbf{x}]$ , that is, when considered as a polynomial on the variables  $\mathbf{x}$ , and regarding  $y$  as a constant. More explicitly,  $H_m^{\mathbf{x}}[P]$  is equal to the sum of all monomials of  $P$  that have degree  $m$  in  $x_1, \dots, x_n$ , without any restrictions on the degree of  $y$ . We also define  $H_{\leq m}^{\mathbf{x}}[P] \equiv \sum_{i=0}^m H_i^{\mathbf{x}}[P]$ .

For example, if  $P(\mathbf{x}, y) \equiv (x_1 x_3 x_4 - x_2^3 + x_1 x_2) y^2 + (x_1^2 x_3 - x_4) y + x_2^2 x_3 - x_1 x_4$ , we have that  $H_3^{\mathbf{x}}[P(\mathbf{x}, y)] \equiv (x_1 x_3 x_4 - x_2^3) y^2 + x_1^2 x_3 y + x_2^2 x_3$ .

Notice that if  $P(\mathbf{x}, y) \equiv \sum_{i=0}^r P_i(\mathbf{x}) y^i$ , then the partial homogeneous parts satisfy the following property:

$$H_m^{\mathbf{x}}[P(\mathbf{x}, y)] \equiv \sum_{i=0}^r H_m^{\mathbf{x}}[P_i(\mathbf{x})] \cdot y^i.$$

Therefore, this definition of partial homogeneous parts agrees with the definition of homogeneous parts if  $P(\mathbf{x}, y)$  does not depend on variable  $y$ .

When talking about partial homogeneous parts of a polynomial, it is useful to have a notion of minimum degree with respect to some variables.

► **Definition 3.2** (Minimum Degree). Let  $f(\mathbf{x}, y) \in \mathbb{F}[\mathbf{x}, y]$  be a polynomial. We define  $\text{mindeg}_{\mathbf{x}}(f(\mathbf{x}, y))$  to be the minimum degree of polynomial  $f(\mathbf{x}, y)$  on the variables  $\mathbf{x}$ . In other words, we have  $\text{mindeg}_{\mathbf{x}}(f(\mathbf{x}, y)) = \min_{\ell} (H_{\ell}^{\mathbf{x}}[f] \neq 0)$ . For instance, if  $f(\mathbf{x}, y) = x_1 x_2 x_3 y - x_1^2 x_2^2 + x_3^5$ , we have that  $\text{mindeg}_{\mathbf{x}}(f) = 3$ .

## 3.2 Basic Operations on Circuits and Formulas

We begin with the following standard lemma on obtaining the homogeneous components of a polynomial. The version below is from [2].

► **Lemma 3.3** (Homogeneous Components Through interpolation). *Let  $P(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$  be a polynomial with degree  $\deg(P) = m$  such that  $P$  can be computed by a formula (circuit)  $\Gamma$  of depth  $d$ . Then, there exists a formula (circuit)  $\Delta$  with  $m + 1$  outputs, of size  $|\Delta| \leq 9m^2 \cdot |\Gamma|$  and depth  $\text{depth}(\Delta) \leq \text{depth}(\Gamma) + 1$  that computes  $H_0^{\mathbf{x}}[P], \dots, H_m^{\mathbf{x}}[P]$ . Moreover, if the topmost gate in the formula (circuit) for  $P(\mathbf{x})$  is an addition gate, then we have  $\text{depth}(\Delta) = \text{depth}(\Gamma) = d$ .*

The next lemma shows us how to obtain the coefficients of a polynomial through interpolation.



► **Lemma 3.4** (Interpolation). *Let  $P(\mathbf{x}, y) \equiv \sum_{i=0}^r y^i P_i(\mathbf{x})$  be a polynomial computed by a formula (circuit)  $\Gamma$ . Then for each  $i \in \{0, 1, \dots, r\}$ , there exists a formula (circuit)  $\Phi_i$  such that  $|\Phi_i| \leq 3r \cdot |\Gamma|$  and  $\Phi_i$  computes the polynomial  $P_i(\mathbf{x})$ .*

Given an irreducible polynomial  $g(\mathbf{x}, y)$  and a polynomial  $P(\mathbf{x}, y)$  that is divisible by  $g$ , it will be useful for us to find a polynomial  $D(\mathbf{x}, y)$  that is divisible by  $g$  and it is also square-free with respect to  $g$ , that is,  $g(\mathbf{x}, y) \nmid \frac{\partial D}{\partial y}(\mathbf{x}, y)$ . The next lemma shows that we can find such a polynomial efficiently.

► **Lemma 3.5.** *Let  $g(\mathbf{x}, y) \in \mathbb{F}[\mathbf{x}, y]$  be an irreducible polynomial that divides a polynomial  $P(\mathbf{x}, y) \in \mathbb{F}[\mathbf{x}, y]$ , where  $\deg_y(P) \leq r$  and let  $\Gamma$  be a formula computing  $P(\mathbf{x}, y)$ . Then, there exists a formula  $\Delta$  that computes a polynomial  $D(\mathbf{x}, y)$  such that  $g(\mathbf{x}, y) \mid D(\mathbf{x}, y)$ ,  $g(\mathbf{x}, y) \nmid \frac{\partial D}{\partial y}(\mathbf{x}, y)$ ,  $|\Delta| \leq 9r^2 \cdot |\Gamma|$  and  $\text{depth}(\Delta) \leq \text{depth}(\Gamma)$ . Moreover, the output gate of  $\Delta$  is an addition gate and for each variable  $z \in \{\mathbf{x}, y\}$ , we have that  $\deg_z(D) \leq \deg_z(P)$ .*

The following observation will be very useful to convert small depth formulas into formulas with fanin bounded by 2.

► **Observation 3.6.** *Any formula  $\Phi$  of size  $s$  and depth  $d$ , without restrictions on the fanin of any of its gates, can be computed by a formula  $\Psi$  of size  $2s$  and depth  $d \cdot (1 + \log(s))$ , where each gate has fanin 2.*

To see that this observation is true, just replace each addition (multiplication) gate of fanin  $t$  by a balanced formula of size  $2t$  made only with addition (multiplication) gates. Since  $t \leq s$ , and a balanced formula of size  $2t$  has depth  $1 + \log t$ , we have that each gate will be replaced by a formula of depth at most  $1 + \log s$ . The replacement by a balanced formula clearly does not change the computation, and the depth increases by a multiplicative factor of  $1 + \log s$ , as we wanted.

### 3.3 Reversal of Polynomials

In this section, we define a very useful operation for polynomials, which serves as a crucial tool in the proof of our main theorem. This operation, which we call *reversal*, simply maps a polynomial  $P(\mathbf{x}, y) \equiv \sum_{i=0}^r P_i(\mathbf{x})y^i$ , with  $P_r(\mathbf{x}) \cdot P_0(\mathbf{x}) \neq 0$ , to  $\tilde{P}(\mathbf{x}) \equiv \sum_{i=0}^r P_i(\mathbf{x})y^{r-i}$ .

The restriction that  $P_r(\mathbf{x}) \cdot P_0(\mathbf{x}) \neq 0$  is needed in this paper because it preserves irreducibility, as we will see in Lemma 3.8 and Corollary 3.9. We begin by showing that the reversal can be computed almost as efficiently as the original polynomial.

► **Lemma 3.7** (Reversal Lemma). *Let  $P(\mathbf{x}, y) \equiv \sum_{i=0}^r y^i P_i(\mathbf{x})$  be a polynomial computed by a formula (circuit)  $\Gamma$ , where  $P_r(\mathbf{x}) \cdot P_0(\mathbf{x}) \neq 0$ . Let  $\tilde{P}(\mathbf{x}, y) \equiv \sum_{i=0}^r y^{r-i} P_i(\mathbf{x})$  be its reversal. There exists a formula (circuit)  $\Delta$  computing  $\tilde{P}$  such that  $|\Delta| = 8r^2 \cdot |\Gamma|$ .*

We now connect the reversal operation to divisibility and irreducibility of polynomials.

► **Lemma 3.8** (Divisibility with Reversals). *Let  $P(\mathbf{x}, y) \equiv \sum_{i=0}^r y^i P_i(\mathbf{x})$ , with  $P_r(\mathbf{x}) \cdot P_0(\mathbf{x}) \neq 0$  and  $f(\mathbf{x}, y) \equiv \sum_{i=0}^k y^i f_i(\mathbf{x})$ , with  $f_k(\mathbf{x}) \cdot f_0(\mathbf{x}) \neq 0$ , be two polynomials. In addition, let*

$\tilde{P}(\mathbf{x}, y) \equiv \sum_{i=0}^r y^{r-i} P_i(\mathbf{x})$  and  $\tilde{f}(\mathbf{x}, y) \equiv \sum_{i=0}^k y^{k-i} f_i(\mathbf{x})$  be their reversals. Then, we have that

$$f \mid P \iff \tilde{f} \mid \tilde{P}.$$

Since divisibility is preserved by taking reversals, we have the following corollary:

► **Corollary 3.9** (Irreducibility of Reversals). Let  $P(\mathbf{x}, y) \equiv \sum_{i=0}^r y^i P_i(\mathbf{x})$ , with  $P_r(\mathbf{x}) \cdot P_0(\mathbf{x}) \neq 0$ ,

be an irreducible polynomial in  $\mathbb{F}[\mathbf{x}, y]$ . In addition, let  $\tilde{P}(\mathbf{x}, y) \equiv \sum_{i=0}^r y^{r-i} P_i(\mathbf{x})$  be its reversal.

Then, we have that

$$P \text{ is irreducible} \iff \tilde{P} \text{ is irreducible.}$$

Another useful property of reversals is that if two univariate polynomials do not share a common root, then their reversals will not share any root either. This gives us the following lemma:

► **Lemma 3.10.** If  $f(x), g(x) \in \mathbb{F}[x]$  do not share any common roots, then their reversals  $\tilde{f}(x), \tilde{g}(x)$  do not share any roots either.

## 4 Properly Splitting Variable Restrictions

In this section, we study properties of pairs of polynomials  $f(\mathbf{x}, y), g(\mathbf{x}, y)$  which share no common factor involving the variable  $y$ . We state a lemma on restrictions of the  $\mathbf{x}$  variables of  $f$  and  $g$  that preserve the property that their restrictions share no common factors in  $y$ . We denote such restrictions as *properly splitting* variable restrictions.

► **Definition 4.1** (Properly Splitting Restrictions). Let  $\mathbf{x} = (x_1, \dots, x_n)$ , where  $n \geq 1$ , and let  $f(\mathbf{x}, y) \in \mathbb{F}[\mathbf{x}, y]$  be an irreducible polynomial such that  $\deg_y(f) \geq 1$ . In addition, let  $g(\mathbf{x}, y) \in \mathbb{F}[\mathbf{x}, y]$  be a polynomial with  $\deg_y(g) \geq 1$  that is not divisible by  $f(\mathbf{x}, y)$ . We say that  $\mathbf{c} \in \mathbb{F}^n$  *properly splits*  $f(\mathbf{x}, y)$  with respect to  $g(\mathbf{x}, y)$  if the following conditions hold:

1.  $f(\mathbf{c}, y)$  is a polynomial with exactly  $\deg_y(f)$  *distinct* roots in  $\mathbb{F}$  and
2.  $f(\mathbf{c}, y)$  and  $g(\mathbf{c}, y)$  share no common roots.

With the definition above, we are now ready to state the main lemma of this section. This lemma tells us that the set of restrictions that properly split an irreducible polynomial  $f(\mathbf{x}, y)$  with respect to a polynomial  $g(\mathbf{x}, y)$  that is not divisible by  $f(\mathbf{x}, y)$  is the complement of an algebraic set. This implies that a random restriction of the variables  $\mathbf{x}$  will properly split  $f(\mathbf{x}, y)$  with respect to  $g(\mathbf{x}, y)$ .

► **Lemma 4.2.** Let  $\mathbf{x} = (x_1, \dots, x_n)$ , where  $n \geq 1$  and  $f(\mathbf{x}, y) \in \mathbb{F}[\mathbf{x}, y]$  be an irreducible polynomial such that  $\deg_y(f) \geq 1$ . In addition, let  $g(\mathbf{x}, y) \in \mathbb{F}[\mathbf{x}, y]$  be a polynomial with  $\deg_y(g) \geq 1$  that is not divisible by  $f(\mathbf{x}, y)$ . Then, there exists a nonzero polynomial  $G(\mathbf{x})$  with  $\deg(G) \leq 2 \deg(f)^2 + 2 \deg(f) \deg(g)$  for which the following holds: for any value  $\mathbf{c} \in \mathbb{F}^n$  such that  $G(\mathbf{c}) \neq 0$ , we have that  $\mathbf{c}$  properly splits  $f(\mathbf{x}, y)$  with respect to  $g(\mathbf{x}, y)$ .

## 5 Standard Forms and Approximation Polynomials

In this section we define the notion of standard forms in  $\mathbb{F}(\mathbf{x})[y]$ , that is, the ring of polynomials on the variable  $y$  with coefficients being rational functions on the variables  $\mathbf{x}$ .

We also define the approximation polynomial of a standard form. These concepts will be useful when factoring a polynomial  $P(\mathbf{x}, y) \in \mathbb{F}[\mathbf{x}, y]$ , since our factorization procedure will use standard forms to obtain the factors of  $P(\mathbf{x}, y)$  that depend of the variable  $y$ . We begin with the following definition:

► **Definition 5.1** (Standard Form and Approximation Polynomials). We say that  $f(\mathbf{x}, y) \in \mathbb{F}(\mathbf{x})[y]$  is in *standard form* if

$$f(\mathbf{x}, y) \equiv \frac{1}{1 - g(\mathbf{x})} \cdot \sum_{i=0}^k f_i(\mathbf{x})y^i,$$

where  $f_i(\mathbf{x}), g(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$ ,  $f_k(\mathbf{x}) \neq 0$  and  $g(\mathbf{0}) = 0$ . Moreover, we will say that  $f$  is in *monic standard form* if  $f_k(\mathbf{x}) \equiv 1 - g(\mathbf{x})$ . For a given parameter  $m \in \mathbb{N}$ , we define the *approximation polynomial* of the standard form  $f$  to degree  $m$ , as the polynomial  $\psi_{f,m}(\mathbf{x}, y) \in \mathbb{F}[\mathbf{x}, y]$  given by

$$\psi_{f,m}(\mathbf{x}, y) = (1 + g(\mathbf{x}) + \dots + g(\mathbf{x})^m) \cdot \sum_{i=0}^k f_i(\mathbf{x})y^i.$$

In order to state some useful properties of approximation polynomials, we will need to extend the definition of reversals to standard forms.

► **Definition 5.2.** Let  $f(\mathbf{x}, y)$  be a standard form as above, with the additional condition that  $f_0(\mathbf{x}) \neq 0$ . We define the reversal of  $f(\mathbf{x}, y)$  as the following standard form:

$$\tilde{f}(\mathbf{x}, y) \equiv \frac{1}{1 - g(\mathbf{x})} \cdot \sum_{i=0}^k f_i(\mathbf{x})y^{k-i}.$$

The following observations about standard forms reveal much of its usefulness when factoring a polynomial.

► **Observation 5.3.** If  $f(\mathbf{x}, y) \in \mathbb{F}(\mathbf{x})[y]$  is in standard form as above, notice that the following holds for all  $m \in \mathbb{N}$ :

1.  $H_{\leq m}^{\mathbf{x}}[(1 - g(\mathbf{x})) \cdot \psi_{f,m}(\mathbf{x}, y)] \equiv H_{\leq m}^{\mathbf{x}}[(1 - g(\mathbf{x})) \cdot f(\mathbf{x}, y)]$ .
2. If  $m \geq \deg((1 - g(\mathbf{x})) \cdot f(\mathbf{x}, y))$ , we have:

$$H_{\leq m}^{\mathbf{x}}[(1 - g(\mathbf{x})) \cdot \psi_{f,m}(\mathbf{x}, y)] \equiv (1 - g(\mathbf{x})) \cdot f(\mathbf{x}, y).$$

3.  $H_{\leq m}^{\mathbf{x}}[\psi_{\tilde{f},m}(\mathbf{x}, y)] \equiv H_{\leq m}^{\mathbf{x}}[\tilde{\psi}_{f,m}(\mathbf{x}, y)]$ .
4. If  $h(\mathbf{x}, y) \equiv f(\mathbf{x}, y + \gamma)$ , where  $\gamma \in \mathbb{F}$ , we have that  $h(\mathbf{x}, y)$  is also a standard form and

$$H_{\leq m}^{\mathbf{x}}[\psi_{f,m}(\mathbf{x}, y + \gamma)] \equiv H_{\leq m}^{\mathbf{x}}[\psi_{h,m}(\mathbf{x}, y)].$$

## 6 Approximating the Roots of a Polynomial

In this section, we proceed in a similar way as in [2] and find approximations of the roots of a polynomial  $P(\mathbf{x}, y)$  up to degree  $t$ . That is, as we defined in the introduction, we find polynomials  $q_t(\mathbf{x})$  such that  $H_{\leq t}^{\mathbf{x}}[P(\mathbf{x}, q_t(\mathbf{x}))] \equiv 0$ . Moreover, we observe that under certain conditions on the polynomial  $P(\mathbf{x}, y)$  these roots are well-defined and unique given their constant coefficient. This uniqueness condition will be useful because it will allow us to construct any factor of  $P(\mathbf{x}, y)$  through the lifting procedure, since a factor  $f(\mathbf{x}, y)$  of  $P(\mathbf{x}, y)$  will share some of the roots of  $P(\mathbf{x}, y)$ . We begin with the approximation lemma:

► **Lemma 6.1** (Approximation Lemma). *Let  $P(\mathbf{x}, y) \in \mathbb{F}[\mathbf{x}, y]$ ,  $P'(\mathbf{x}, y) \equiv \frac{\partial P}{\partial y}(\mathbf{x}, y)$  and  $\mu \in \mathbb{F}$  be such that  $P(\mathbf{0}, \mu) = 0$  but  $P'(\mathbf{0}, \mu) = \xi \neq 0$ . Then, for each  $t \geq 0$ , there exists a **unique** polynomial  $q_t(\mathbf{x})$  s.t.  $\deg(q_t) \leq t$ ,  $q_t(\mathbf{0}) = \mu$  and*

$$H_{\leq t}^{\mathbf{x}}[P(\mathbf{x}, q_t(\mathbf{x}))] \equiv 0.$$

Moreover, if  $P$  can be computed by a formula (circuit)  $\Gamma$  such that its output gate is an addition gate, there is a formula (circuit)  $\Phi_t$  for the polynomial  $q_t(\mathbf{x})$  such that the output gate of  $\Phi_t$  is an addition gate,  $\text{depth}(\Phi_t) \leq \text{depth}(\Gamma) + 2$  and

$$|\Phi_t| \leq 200(tr)^2 \binom{t+r+1}{r+1} \cdot |\Gamma|.$$

If we require the fanin of the formula (circuit) to be 2, then the size of  $\Phi_t$  does not change, and  $\text{depth}(\Phi_t) \leq \text{depth}(\Gamma) + 5r \log(t)$ .

Now that we know that any root of a polynomial  $P(\mathbf{x}, y)$  of small individual degree computed by a small formula can be approximated by a small formula, the next corollary uses the uniqueness of the approximation of the root to show that the same is true for any factor of  $P(\mathbf{x}, y)$ .

► **Corollary 6.2.** *Let  $P(\mathbf{x}, y)$  and  $\mu \in \mathbb{F}$  be defined as in Lemma 6.1 and for each  $t \in \mathbb{N}_0$ , let  $q_t(\mathbf{x})$  be the unique polynomial obtained from Lemma 6.1. If  $h(\mathbf{x}, y) \in \mathbb{F}[\mathbf{x}, y]$  is such that  $h(\mathbf{0}, \mu) = 0$ ,  $\frac{\partial h}{\partial y}(\mathbf{0}, \mu) \neq 0$  and there exist  $t \in \mathbb{N}$  and  $Q(\mathbf{x}, y) \in \mathbb{F}[\mathbf{x}, y]$  such that*

$$H_{\leq t}^{\mathbf{x}}[P(\mathbf{x}, y)] \equiv H_{\leq t}^{\mathbf{x}}[h(\mathbf{x}, y) \cdot Q(\mathbf{x}, y)], \quad (1)$$

then the polynomial  $q_t(\mathbf{x})$  also satisfies

$$H_{\leq t}^{\mathbf{x}}[h(\mathbf{x}, q_t(\mathbf{x}))] \equiv 0, \quad \forall t \geq 0.$$

## 7 Proof of the Main Theorem

In this section, we give the proof of our main theorem. In addition, we state the consequences of the main theorem for both small formula size and depth of circuits computing factors of polynomials with small bounded degree.

► **Lemma 7.1** (Main Lemma). *Let  $P(\mathbf{x}, y) \in \mathbb{F}[\mathbf{x}, y]$  be such that  $\deg_y(P) = r$ , and also  $\deg_{x_i}(P) \leq r, \forall i \in \{1, \dots, n\}$ . Let  $P'(\mathbf{x}, y) \equiv \frac{\partial P}{\partial y}(\mathbf{x}, y)$ . In addition, let  $f(\mathbf{x}, y) \in \mathbb{F}(\mathbf{x})[y]$  be in monic standard form and assume it is irreducible over  $\mathbb{F}(\mathbf{x})[y]$ , satisfying the following conditions:*

1.  $f(\mathbf{x}, y) \mid P(\mathbf{x}, y)^1$ ,
2.  $f(\mathbf{0}, y)$  has exactly  $\deg_y(f)$  distinct roots<sup>2</sup>,
3.  $P'(\mathbf{0}, y)$  and  $f(\mathbf{0}, y)$  share no common roots.

If there exists a formula (circuit)  $\Gamma$  computing  $P$  with output gate being an addition gate,  $|\Gamma| = s$  and  $\text{depth}(\Gamma) = d$ , then for every  $m \geq 1$ , there exist formulas (circuits)  $\Psi_m$  and  $\tilde{\Psi}_m$  with each output gate being a multiplication gate, of size

$$\max(|\Psi_m|, |\tilde{\Psi}_m|) \leq 300m^2r^3 \cdot \binom{m+r+1}{r+1} \cdot s$$

<sup>1</sup> Since  $P(\mathbf{x}, y) \in \mathbb{F}[\mathbf{x}, y]$ , this condition is equivalent to the existence of  $Q(\mathbf{x}, y) \in \mathbb{F}[\mathbf{x}, y]$  such that  $f(\mathbf{x}, y) \cdot Q(\mathbf{x}, y) \equiv P(\mathbf{x}, y)$ .

<sup>2</sup> Note that we can evaluate  $f(\mathbf{x}, y)$  at  $\mathbf{x} = \mathbf{0}$ , since  $f(\mathbf{x}, y)$  is in standard form.

and depth  $\max(\text{depth}(\Psi_m), \text{depth}(\tilde{\Psi}_m)) \leq d + 3$  such that

$$H_{\leq m}^{\mathbf{x}}[\Psi_m] \equiv H_{\leq m}^{\mathbf{x}}[\psi_{f,m}(\mathbf{x}, y)] \quad \text{and}$$

$$H_{\leq m}^{\mathbf{x}}[\tilde{\Psi}_m] \equiv H_{\leq m}^{\mathbf{x}}[\psi_{\tilde{f},m}(\mathbf{x}, y)].$$

If we require the in-degree of the formula (circuit) to be 2, then the size of  $\Psi_m$  or  $\tilde{\Psi}_m$  does not change, and  $\max(\text{depth}(\Psi_m), \text{depth}(\tilde{\Psi}_m)) \leq d + 10r \log m$ .

With the Main Lemma stated above, we are now able to state and prove our main theorem.

► **Theorem 7.2 (Main Theorem).** *Let  $P(\mathbf{x}) \in \mathbb{F}[\mathbf{x}] \setminus \{0\}$  be such that  $\deg_{x_i}(P) \leq r$ ,  $1 \leq i \leq n$ ,  $P(\mathbf{0}) \neq 0$  and let  $\Gamma$  be a formula (circuit) of size  $s$  and depth  $d$  computing  $P$ . Let  $f(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$  be a factor of  $P(\mathbf{x})$ , and let  $m$  be a positive integer. There exists a polynomial  $G(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$  of total degree  $\deg(G) \leq 4r^3n^3$  such that if  $\mathbf{c} \in \mathbb{F}^n$  satisfies  $G(\mathbf{c}) \neq 0$  then there exists a formula  $\Phi_m$  whose output gate is a multiplication gate and for which*

$$\text{depth}(\Phi_m) \leq d + 4^3,$$

$$|\Phi_m| \leq 60000m^2r^8n \cdot \binom{m+r+1}{r+1}^s \quad \text{and}$$

$$H_{\leq m}^{\mathbf{x}}[\Phi_m(\mathbf{x})] \equiv H_{\leq m}^{\mathbf{x}}[f(\mathbf{x} + \mathbf{c})].$$

If we require the in-degree of the formula (circuit) to be 2, then the size of  $\Phi_m$  does not change, and  $\text{depth}(\Phi_m) \leq d + 20r \log m$ .

**Proof.** The proof of the theorem is by induction on the number of variables. The bound is trivial in the univariate case, since if  $f(x), P(x) \in \mathbb{F}[x]$ , where  $\deg(f) = k \leq r$  and  $f \mid P$ , then we can write

$$f(x) = c \cdot \prod_{i=1}^k (x - \mu_i),$$

which can be trivially computed by a formula  $\Psi$  of size  $\leq 50k$  and depth 2. In this case, setting  $G(x)$  to be any constant polynomial, for instance  $G(x) \equiv 1$ ,  $\mathbf{c} = \mathbf{0}$  and  $\Phi_m = \Psi$ , takes care of the base case.

Hence, let's assume that the claim is true for polynomials  $P(\mathbf{x}) \in \mathbb{F}[\mathbf{x}] = \mathbb{F}[x_1, \dots, x_n]$  with  $P(\mathbf{0}) \neq 0$ , for some  $n \geq 1$ . Now we will prove that the same bounds hold for polynomials  $P(\mathbf{x}, y) \in \mathbb{F}[\mathbf{x}, y]$  s.t.  $P(\mathbf{0}, 0) \neq 0$ . Let  $P(\mathbf{x}, y) \in \mathbb{F}[\mathbf{x}, y]$  be a polynomial computed by  $\Gamma$  and  $f(\mathbf{x}, y) \in \mathbb{F}[\mathbf{x}, y]$  be a factor of  $P(\mathbf{x}, y)$ . We can assume that  $f(\mathbf{x}, y)$  and  $P(\mathbf{x}, y)$  depend on  $y$ , otherwise we can simply restrict the formula  $\Gamma$  to  $\Gamma|_{y=0}$ , and by the induction hypothesis the result follows.

Let

$$P(\mathbf{x}, y) \equiv \sum_{i=0}^r C_i(\mathbf{x})y^i, \quad \text{and} \quad f(\mathbf{x}, y) \equiv q(\mathbf{x}) \cdot \prod_{i=1}^t f_i(\mathbf{x}, y)^{e_i}, \quad \text{with}$$

$$f_i(\mathbf{x}, y) \equiv \sum_{j=0}^{k_i} f_{ij}(\mathbf{x})y^j, \quad \text{where } f_{i0}(\mathbf{x}) \cdot f_{ik_i}(\mathbf{x}) \neq 0, \quad \forall 1 \leq i \leq t.$$

<sup>3</sup> If the bottom gates are addition gates, then the depth is bounded by  $d + 3$ .

where each  $f_i(\mathbf{x}, y) \in \mathbb{F}[\mathbf{x}, y]$  is an irreducible polynomial. Since  $P(\mathbf{0}, 0) \neq 0$ , we have that  $C_0(\mathbf{x}) \equiv P(\mathbf{x}, 0) \neq 0$ , and moreover, that  $C_0(\mathbf{0}) \neq 0$ . Let

$$u(\mathbf{x}) \equiv f(\mathbf{x}, 0) \equiv q(\mathbf{x}) \cdot \prod_{i=1}^t f_{i0}(\mathbf{x})^{e_i}.$$

Notice that  $f(\mathbf{x}, y) \mid P(\mathbf{x}, y) \Rightarrow u(\mathbf{x}) \mid C_0(\mathbf{x})$ . In addition, notice that  $C_0(\mathbf{0}) \neq 0$  and  $C_0(\mathbf{x})$  can be computed by the formula  $\Gamma|_{y=0}$ , where  $|\Gamma|_{y=0}| \leq |\Gamma|$  and  $\text{depth}(\Gamma|_{y=0}) \leq \text{depth}(\Gamma)$ . Therefore, by induction hypothesis, there exists  $H(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$  with  $\deg(H) \leq 4r^3n^3$  such that for any  $\mathbf{a} \in \mathbb{F}^n$  for which  $H(\mathbf{a}) \neq 0$ , there exists a formula  $\Lambda_m$  with output gate being a multiplication gate, such that

$$\text{depth}(\Lambda_m) \leq d + 4,$$

$$|\Lambda_m| \leq 60000m^2r^8n \cdot \binom{m+r+1}{r+1}^s \quad \text{and}$$

$$H_{\leq m}^{\mathbf{x}}[\Lambda_m(\mathbf{x})] \equiv H_{\leq m}^{\mathbf{x}}[u(\mathbf{x} + \mathbf{a})].$$

Now that we have an approximation to the factor  $u(\mathbf{x})$ , which is the constant term of the polynomial  $f(\mathbf{x}, y)$  when seen as a polynomial in the variable  $y$ , we want to use Lemma 7.1 to find the factors of  $f(\mathbf{x}, y)$  that contain  $y$ . For this, we will first need to find polynomials  $D_i(\mathbf{x}, y)$  with small formulas such that  $f_i(\mathbf{x}, y) \mid D_i(\mathbf{x}, y)$  and each  $D_i$  is square-free with respect to  $f_i(\mathbf{x}, y)$ .

Fortunately, Lemma 3.5 tells us that for each (irreducible) polynomial  $f_i(\mathbf{x}, y)$ , we can find formulas  $\Delta_i$  of size  $\leq 9r^2|\Gamma|$  computing polynomials  $D_i(\mathbf{x}, y)$  such that  $\deg_{x_j}(D_i) \leq r$ ,  $1 \leq j \leq n$ ,  $\deg_y(D_i) \leq r$ ,  $f_i(\mathbf{x}, y) \mid D_i(\mathbf{x}, y)$  but  $f_i(\mathbf{x}, y) \nmid \frac{\partial D_i}{\partial y}(\mathbf{x}, y)$ . Moreover these formulas have an addition gate as output gate.

Since  $f_i(\mathbf{x}, y)$  is irreducible with  $\deg_y(f_i) \geq 1$  and  $f_i(\mathbf{x}, y) \nmid \frac{\partial D_i}{\partial y}(\mathbf{x}, y)$ , Lemma 4.2 implies that there exists a polynomial  $G_i(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$  with

$$\deg(G_i) \leq 2 \deg(f_i)^2 + 2 \deg(f_i) \deg(D_i) \leq 4r^2n^2$$

such that for any  $\mathbf{c} \in \mathbb{F}^n$  where  $G_i(\mathbf{c}) \neq 0$  we have that  $\mathbf{c}$  properly splits  $f_i(\mathbf{c}, y)$  with respect to  $\frac{\partial D_i}{\partial y}(\mathbf{c}, y)$ .

Let

$$G(\mathbf{x}, y) \equiv H(\mathbf{x}) \cdot C_0(\mathbf{x}) \cdot \prod_{i=1}^t G_i(\mathbf{x}) \quad \text{and} \quad (\mathbf{c}, \gamma) \in \mathbb{F}^{n+1} \quad \text{be s.t.} \quad G(\mathbf{c}, \gamma) \neq 0.^4$$

---

<sup>4</sup> At first, it may seem strange that  $G(\mathbf{x}, y)$  does not depend on the variable  $y$ , since if we continued this argument by induction we would arrive at the conclusion that  $G(\mathbf{x}, y)$  is the constant polynomial. However, notice that even though  $H(\mathbf{x})$  does not depend on the variable  $x_n$ , the polynomial  $G(\mathbf{x}, y)$  depends on  $x_n$ , since the polynomials  $C_0(\mathbf{x})$  and  $G_i(\mathbf{x})$  depend on  $x_n$ . The right way to see this dependence is the following:  $G(\mathbf{x}, y)$  depends on every variable except the variable used by the lifting procedure, which in this case is the variable  $y$ . Hence, we will have that  $H(\mathbf{x})$  depends on all the variables except  $x_n$  (if we choose to perform the lifting with respect to  $x_n$ ).

Denote

$$Q(\mathbf{x}, y) \equiv P(\mathbf{x} + \mathbf{c}, y) \equiv \sum_{i=0}^r Q_i(\mathbf{x})y^i, \quad h_i(\mathbf{x}, y) \equiv f_i(\mathbf{x} + \mathbf{c}, y) \equiv \sum_{j=0}^{k_i} h_{ij}(\mathbf{x})y^j \quad \text{and}$$

$$h(\mathbf{x}, y) \equiv f(\mathbf{x} + \mathbf{c}, y) \equiv q(\mathbf{x} + \mathbf{c}) \cdot \prod_{i=1}^t h_i(\mathbf{x}, y)^{e_i}.$$

Since  $h_{i0}(\mathbf{x}) \equiv f_{i0}(\mathbf{x} + \mathbf{c}, 0) \mid P(\mathbf{x} + \mathbf{c}, 0) \equiv C_0(\mathbf{x} + \mathbf{c})$  and  $C_0(\mathbf{c}) \neq 0$  (because  $G(\mathbf{c}, \gamma) \neq 0$ ), we have that  $h_{i0}(\mathbf{0}) \neq 0$ , for all  $1 \leq i \leq t$ . Hence, after normalization by a proper field element, we can write each  $h_{i0}$  in the following form:

$$h_{i0}(\mathbf{x}) = 1 - g_i(\mathbf{x}), \quad \text{where } g_i(\mathbf{0}) \equiv 0.$$

In addition, notice that  $f_{ik_i}(\mathbf{x}) \neq 0 \Rightarrow h_{ik_i}(\mathbf{x}) \equiv f_{ik_i}(\mathbf{x} + \mathbf{c}) \neq 0$ .

Moreover, notice that  $f_i(\mathbf{x}, y)$  is irreducible with  $f_{i0}(\mathbf{x}) \cdot f_{ik_i}(\mathbf{x}) \neq 0$  implies that  $h_i(\mathbf{x}, y)$  is irreducible with  $h_{i0}(\mathbf{x}) \cdot h_{ik_i}(\mathbf{x}) \neq 0$ , which implies (by Corollary 3.9) that the polynomial  $\tilde{h}_i(\mathbf{x}, y) \equiv \sum_{j=0}^{k_i} h_{ij}(\mathbf{x})y^{k_i-j}$  is irreducible in  $\mathbb{F}[\mathbf{x}, y]$ . Hence, we have that  $\ell_i(\mathbf{x}, y) \equiv \frac{\tilde{h}_i(\mathbf{x}, y)}{h_{i0}(\mathbf{x})}$  is a monic irreducible standard form in  $\mathbb{F}(\mathbf{x})[y]$ .

Because  $f_i(\mathbf{x}, y) \mid D_i(\mathbf{x}, y)$  and  $f_i(\mathbf{x}, y) \nmid \frac{\partial D_i}{\partial y}(\mathbf{x}, y)$ , by Lemma 3.8 we obtain that  $h_i(\mathbf{x}, y) \mid E_i(\mathbf{x}, y) \equiv D_i(\mathbf{x} + \mathbf{c}, y)$  and  $h_i(\mathbf{x}, y) \nmid \frac{\partial E_i}{\partial y}(\mathbf{x}, y) \equiv \frac{\partial D_i}{\partial y}(\mathbf{x} + \mathbf{c}, y)$ .

Since  $h_i(\mathbf{0}, y) \equiv f_i(\mathbf{c}, y)$ , we also have that  $h_i(\mathbf{0}, y)$  has no common roots with  $\frac{\partial E_i}{\partial y}(\mathbf{0}, y)$ . The following claim shows that  $\ell_i(\mathbf{x}, y)$  satisfies the conditions of Lemma 7.1.

► **Claim 7.3.** *For each  $i \in \{1, \dots, t\}$ , the monic irreducible standard form  $\ell_i(\mathbf{x}, y) \equiv \frac{\tilde{h}_i(\mathbf{x}, y)}{h_{i0}(\mathbf{x})}$  and the polynomial  $\tilde{E}_i(\mathbf{x}, y)$  satisfy the conditions of Lemma 7.1.*

**Proof of claim.** Notice that conditions (i) and (ii) from Lemma 7.1 follow from the fact that  $h_i(\mathbf{x}, y) \mid E_i(\mathbf{x}, y)$  and Lemmas 3.8 and 4.2. Condition (iii) follows from the fact that  $\frac{h_i(\mathbf{0}, y)}{h_{i0}(\mathbf{0})} \equiv h_i(\mathbf{0}, y)$  shares no common roots with  $\frac{\partial E_i}{\partial y}(\mathbf{0}, y)$  and from Lemma 3.10.

This finishes the proof of the claim. ◀

Now that we have rational functions in monic standard form that are, in a certain sense, computing the reversal of each  $f_i(\mathbf{x}, y)$ , we can use the main lemma to lift the factorization of the approximation polynomial of  $f_i(\mathbf{x}, y)/f_{i0}(\mathbf{x})$ .<sup>5</sup>

Since each  $\ell_i(\mathbf{x}, y)$  and  $\tilde{E}_i(\mathbf{x}, y)$  satisfy the conditions of Lemma 7.1, and  $\tilde{E}_i(\mathbf{x}, y)$  can be computed by a formula  $\Upsilon_i$  of size  $|\Upsilon_i| \leq 180r^4 \cdot |\Gamma| = 180r^4s$  and depth  $\text{depth}(\Upsilon_i) \leq d + 1$  (since  $\Upsilon_i$  is a shift of  $\tilde{\Delta}_i$ ), we have that there exists a formula  $\Psi_{i,m}$  having as output gate a multiplication gate,  $\text{depth}(\Psi_{i,m}) \leq \text{depth}(\Upsilon_i) + 3 \leq d + 4$  and size

$$|\Psi_{i,m}| \leq 300m^2r^3 \cdot \binom{m+r+1}{r+1} \cdot 180r^4 \cdot s \leq 60000m^2r^7 \cdot \binom{m+r+1}{r+1} \cdot s$$

<sup>5</sup> In actuality, we are performing a lift of a shift of  $f_i(\mathbf{x}, y)$ .

such that

$$H_{\leq m}^{\mathbf{x}}[\Psi_{i,m}] \equiv H_{\leq m}^{\mathbf{x}}[\psi_{\tilde{\ell}_i(\mathbf{x},y),m}^{\sim}(\mathbf{x},y)].$$

By Observation 5.3, we have that

$$\begin{aligned} H_{\leq m}^{\mathbf{x}}[h_{i0}(\mathbf{x}) \cdot \psi_{\tilde{\ell}_i(\mathbf{x},y),m}^{\sim}(\mathbf{x},y)] &\equiv H_{\leq m}^{\mathbf{x}}[\tilde{\ell}_i(\mathbf{x},y) \cdot h_{i0}(\mathbf{x})] \equiv H_{\leq m}^{\mathbf{x}}[h_i(\mathbf{x},y)], \quad \text{and also} \\ H_{\leq m}^{\mathbf{x}}[h_{i0}(\mathbf{x}) \cdot \psi_{\tilde{\ell}_i(\mathbf{x},y),m}^{\sim}(\mathbf{x},y+\gamma)] &\equiv H_{\leq m}^{\mathbf{x}}[h_i(\mathbf{x},y+\gamma)]. \end{aligned}$$

In addition, from the formulas  $\Psi_{i,m}$  and from the fact that  $\sum_{i=1}^t e_i \leq r$ , we have that the formula given by  $\Psi_m = \prod_{i=1}^t \Psi_{i,m}^{e_i}$  is of size

$$|\Psi_m| \leq \sum_{i=1}^t e_i \cdot |\Psi_{i,m}| \leq r \cdot \max_{1 \leq i \leq t} (|\Psi_{i,m}|) \leq 60000m^2r^8 \cdot \binom{m+r+1}{r+1} \cdot s$$

and computes the following polynomial:

$$H_{\leq m}^{\mathbf{x}}[\Psi_m(\mathbf{x},y)] \equiv H_{\leq m}^{\mathbf{x}} \left[ \prod_{i=1}^t \psi_{\tilde{\ell}_i(\mathbf{x},y),m}^{\sim}(\mathbf{x},y+\gamma)^{e_i} \right].$$

Now that we found a formula computing the approximation polynomials  $\psi_{\tilde{\ell}_i(\mathbf{x},y),m}^{\sim}(\mathbf{x},y+\gamma)$ , we can multiply them by  $h_{i0}(\mathbf{x},y)$  and via Observation 5.3 obtain the polynomials  $h_i(\mathbf{x},y)$ , which are the shifts of  $f_i(\mathbf{x},y)$ . Since  $\Psi_m$  computes all of the approximation polynomials, and  $\Lambda_m$  computes all of the leading coefficients, by combining them we can recover the factor  $f(\mathbf{x},y)$ . This is what we do next.

Multiplying  $\Psi_m$  by  $\Lambda_m$ , we have that the formula  $\Phi_m = \Lambda_m \cdot \Psi_m$  is such that

$$|\Phi_m| \leq |\Lambda_m| + |\Psi_m| \leq 60000m^2r^8(n+1) \cdot \binom{m+r+1}{r+1} \cdot s$$

and

$$\begin{aligned} H_{\leq m}^{\mathbf{x}}[\Phi_m(\mathbf{x},y)] &\equiv H_{\leq m}^{\mathbf{x}}[\Lambda_m \cdot \Psi_m] \equiv H_{\leq m}^{\mathbf{x}} \left[ u(\mathbf{x} + \mathbf{c}) \cdot \prod_{i=1}^t \psi_{\frac{h_i(\mathbf{x},y)}{h_{i0}(\mathbf{x})},m}^{\sim}(\mathbf{x},y+\gamma)^{e_i} \right] \\ &\equiv H_{\leq m}^{\mathbf{x}} \left[ q(\mathbf{x} + \mathbf{c}) \cdot \prod_{i=1}^t f_{i0}(\mathbf{x} + \mathbf{c})^{e_i} \cdot \prod_{i=1}^t \psi_{\frac{h_i(\mathbf{x},y)}{h_{i0}(\mathbf{x})},m}^{\sim}(\mathbf{x},y+\gamma)^{e_i} \right] \\ &\equiv H_{\leq m}^{\mathbf{x}} \left[ q(\mathbf{x} + \mathbf{c}) \cdot \prod_{i=1}^t \left( h_{i0}(\mathbf{x}) \cdot \psi_{\frac{h_i(\mathbf{x},y)}{h_{i0}(\mathbf{x})},m}^{\sim}(\mathbf{x},y+\gamma) \right)^{e_i} \right] \\ &\equiv H_{\leq m}^{\mathbf{x}} \left[ q(\mathbf{x} + \mathbf{c}) \cdot \prod_{i=1}^t h_i(\mathbf{x},y+\gamma)^{e_i} \right] \\ &\equiv H_{\leq m}^{\mathbf{x}} \left[ q(\mathbf{x} + \mathbf{c}) \cdot \prod_{i=1}^t f_i(\mathbf{x} + \mathbf{c},y+\gamma)^{e_i} \right] \equiv H_{\leq m}^{\mathbf{x}}[f(\mathbf{x} + \mathbf{c},y+\gamma)]. \end{aligned}$$

Since

$$\deg(G(\mathbf{x},y)) \leq \deg(H) + \deg(C_0) + \sum_{i=1}^t \deg(G_i) \leq 4r^3n^3 + rn + r \cdot 4r^2n^2 \leq 4r^3(n+1)^3,$$

this finishes the induction, and therefore the proof of the theorem. It is clear from the proof, via Observation 3.6, that if we restrict the in-degree to 2, we obtain the desired bound on the depth.  $\blacktriangleleft$



As a corollary of the main theorem, we obtain:

► **Corollary 7.4** (Small Formula – Restatement of Theorem 1.2). *Let  $P(\mathbf{x}) \in \mathbb{F}[\mathbf{x}] \setminus \{0\}$  be such that  $\deg_{x_i}(P) \leq r$ ,  $1 \leq i \leq n$ , and let  $f(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$  be a factor of  $P$ . If there exists a formula  $\Gamma$  of size  $s$  and depth  $d$  computing  $P$ , then there exists a formula  $\Phi$  of depth  $\text{depth}(\Phi) \leq d + 5$  and size*

$$|\Phi| = O\left(n^3 r^{12} \cdot \binom{nr + r + 1}{r + 1} s\right) = \text{poly}((nr)^r, s)$$

such that

$$\Phi(\mathbf{x}) \equiv f(\mathbf{x}).$$

If we require the in-degree of the formula (circuit) to be 2, then the size of  $\Phi$  does not change, and  $\text{depth}(\Phi) \leq d + 30r \log(nr)$ .

**Proof.** Let  $\mathbf{c} \in \mathbb{F}^n$  be such that  $P(\mathbf{c}) \neq 0$ . Such a  $\mathbf{c}$  exists since  $P(\mathbf{x})$  is nonzero. This implies that  $Q(\mathbf{x}) \equiv P(\mathbf{x} + \mathbf{c})$  is computed by the formula  $\Delta(\mathbf{x}) = \Gamma(\mathbf{x} + \mathbf{c})$ , of size  $\leq 2|\Gamma| = 2s$ ,  $\text{depth}(\Delta) \leq d + 1$  and is such that  $Q(\mathbf{0}) = P(\mathbf{c}) \neq 0$ . Hence, by Theorem 7.2, we have that there exists polynomial  $G(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$  of degree  $\deg(G) \leq 4r^3 n^3$  such that for any  $\mathbf{a} \in \mathbb{F}^n$  for which  $G(\mathbf{a}) \neq 0$ , there is a formula  $\Phi_{nr}$  whose output gate is a multiplication gate for which  $\text{depth}(\Phi_{nr}) \leq \text{depth}(\Delta) + 3 \leq d + 4$ , of size

$$|\Phi_{nr}| \leq 120000(nr)^2 r^8 n \cdot \binom{nr + r + 1}{r + 1} s \text{ and such that}$$

$$H_{\leq nr}^{\mathbf{x}}[\Phi_{nr}(\mathbf{x})] \equiv H_{\leq nr}^{\mathbf{x}}[f(\mathbf{x} + \mathbf{c} + \mathbf{a})] \equiv f(\mathbf{x} + \mathbf{c} + \mathbf{a}), \text{ since } nr \geq \deg(f).$$

By the interpolation Lemma 3.4, we obtain that there exists a formula  $\Phi'$  of size

$$|\Phi'| \leq 9r^2 \cdot |\Phi_{nr}|$$

and  $\text{depth}(\Phi') \leq d + 5$  such that  $\Phi'(\mathbf{x}) \equiv f(\mathbf{x} + \mathbf{c} + \mathbf{a})$ . By shifting the inputs of the formula  $\Phi'$  by  $-(\mathbf{c} + \mathbf{a})$ , we have that the new formula just obtained, call it  $\Phi$ , computes the polynomial  $f(\mathbf{x})$ , as we wanted. It is easy to see that  $\Phi$  has the desired upper bound on its size. It is also clear from the proof that if we restrict the in-degree of the formulas (circuits) to be 2, we obtain the desired bounds on the depth. This finishes the proof. ◀

## 8 Conclusion

Besides solving a question posed by Kopparty et al. [10] and Open Problem 19 in [15] for the class of bounded individual degree polynomials, notice that Lemma 7.1 and Theorem 7.2 also provide a framework to convert formulas (circuits) for the approximate roots of a polynomial into actual formulas (circuits) for factors of the same polynomial. Since Lemma 7.1, and therefore Theorem 7.2, uses the Approximation Lemma (Lemma 6.1) as a black-box, any improvements on Lemma 6.1 would lead to better bounds on the size of the formulas for the factors of the input polynomial. Hence, if one can remove the exponential dependence on the parameter  $r$  (the bound on the individual degrees) in the Approximation Lemma, one can fully solve the questions above. This is the main open question left by this work.

**Acknowledgements.** The author would like to thank his advisor Zeev Dvir for all the helpful discussions and encouragement throughout the course of this work.

---

**References**

---

- 1 Benny Chor and Ronald L. Rivest. A knapsack-type public key cryptosystem based on arithmetic in finite fields. *IEEE Transactions on Information Theory*, 34(5):901–909, 1988.
- 2 Z. Dvir, A. Shpilka, and A. Yehudayoff. Hardness-randomness tradeoffs for bounded depth arithmetic circuits. *SIAM J. on Computing*, 39(4):1279–1293, 2009.
- 3 J. von zur Gathen and J. Gerhard. *Modern computer algebra*. Cambridge University Press, 1999.
- 4 J. Von Zur Gathen and E. Kaltofen. Factoring sparse multivariate polynomials. *Journal of Computer and System Sciences*, 31(2):265–287, 1985.
- 5 V. Guruswami and M. Sudan. Improved decoding of reed-solomon and algebraic-geometry codes. *IEEE Trans. Inf. Theor.*, 45(6):1757–1767, September 2006.
- 6 V. Kabanets and R. Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Computational Complexity*, 13(1-2):1–46, 2004.
- 7 E. Kaltofen. Polynomial-time reductions from multivariate to bi- and univariate integral polynomial factorization. *SIAM J. on computing*, 14(2):469–489, 1985.
- 8 E. Kaltofen. Factorization of polynomials given by straight-line programs. In S. Micali, editor, *Randomness in Computation*, volume 5 of *Advances in Computing Research*, pages 375–412. JAI Press, 1989.
- 9 E. Kaltofen. Polynomial factorization: a success story. In *ISSAC*, pages 3–4, 2003.
- 10 Swastik Kopparty, Shubhangi Saraf, and Amir Shpilka. Equivalence of polynomial identity testing and deterministic multivariate polynomial factorization. In *IEEE 29th Conference on Computational Complexity, CCC 2014, Vancouver, BC, Canada, June 11-13, 2014*, pages 169–180, 2014.
- 11 A. K. Lenstra, H. W. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, 1982.
- 12 Hendrik W Lenstra Jr. Finding small degree factors of lacunary polynomials. *Number theory in progress*, 1:267–276, 1999.
- 13 R. Oliveira. Factors of low individual degree polynomials. <http://www.cs.princeton.edu/~rmo/papers/small-depth-factors.pdf>, 2015.
- 14 A. Shpilka and I. Volkovich. On the relation between polynomial identity testing and finding variable disjoint factors. In *ICALP (1)*, pages 408–419, 2010.
- 15 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.
- 16 Madhu Sudan. Decoding of reed solomon codes beyond the error-correction bound. *Journal of Complexity*, 13(1):180–193, 1997.

# Verifiable Stream Computation and Arthur-Merlin Communication

Amit Chakrabarti<sup>1</sup>, Graham Cormode<sup>2</sup>, Andrew McGregor<sup>3</sup>,  
Justin Thaler<sup>4</sup>, and Suresh Venkatasubramanian<sup>5</sup>

- 1 Department of Computer Science, Dartmouth College, USA
- 2 Department of Computer Science, University of Warwick, UK
- 3 Department of Computer Science, UMass Amherst, USA
- 4 Yahoo Labs, New York, USA
- 5 School of Computing, University of Utah, USA

---

## Abstract

In the setting of streaming interactive proofs (SIPs), a client (verifier) needs to compute a given function on a massive stream of data, arriving online, but is unable to store even a small fraction of the data. It outsources the processing to a third party service (prover), but is unwilling to blindly trust answers returned by this service. Thus, the service cannot simply supply the desired answer; it must *convince* the verifier of its correctness via a short interaction after the stream has been seen.

In this work we study “barely interactive” SIPs. Specifically, we show that two or three rounds of interaction suffice to solve several query problems – including Index, Median, Nearest Neighbor Search, Pattern Matching, and Range Counting – with polylogarithmic space and communication costs. Such efficiency with  $O(1)$  rounds of interaction was thought to be impossible based on previous work.

On the other hand, we initiate a formal study of the limitations of constant-round SIPs by introducing a new hierarchy of communication models called Online Interactive Proofs (OIPs). The online nature of these models is analogous to the streaming restriction placed upon the verifier in an SIP. We give upper and lower bounds that (1) characterize, up to quadratic blowups, every finite level of the OIP hierarchy in terms of other well-known communication complexity classes, (2) separate the first four levels of the hierarchy, and (3) reveal that the hierarchy collapses to the fourth level. Our study of OIPs reveals marked contrasts and some parallels with the classic Turing Machine theory of interactive proofs, establishes limits on the power of existing techniques for developing constant-round SIPs, and provides a new characterization of (non-online) Arthur-Merlin communication in terms of an online model.

**1998 ACM Subject Classification** F.1.3 Complexity Measures and Classes

**Keywords and phrases** streaming interactive proofs, Arthur-Merlin communication complexity

**Digital Object Identifier** 10.4230/LIPIcs.CCC.2015.217

## 1 Introduction

The surging popularity of commercial cloud computing services, and more generally outsourced computations, has revealed compelling new applications for the study of *interactive proofs* with highly restricted *verifiers*. Consider, e.g., a retailer (verifier) who lacks the resources to locally process a massive input (say, the set of all its transactions), but can access a powerful but untrusted cloud service provider (prover), who processes the input on the retailer’s behalf. The verifier must work within the confines of the restrictive *data streaming* paradigm, using



© Amit Chakrabarti, Graham Cormode, Andrew McGregor, Justin Thaler,  
and Suresh Venkatasubramanian;  
licensed under Creative Commons License CC-BY

30th Conference on Computational Complexity (CCC’15).

Editor: David Zuckerman; pp. 217–243



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

only a small amount of working memory. The prover must both answer queries about the input (say, “how many pairs of blue jeans have I ever sold?”), and prove that the answer is correct. We refer to this general scenario as *verifiable data stream computation*.

It is useful to look at this computational scenario as “data stream algorithms with access to a powerful (space-unlimited) prover.” As is well known, most interesting data streaming problems have no nontrivial (i.e., sublinear space) algorithms unless one allows approximation. For instance, given a stream  $\sigma$  of tokens from the universe  $[n] := \{1, 2, \dots, n\}$ , which implicitly defines frequencies  $f_j$  for each  $j \in [n]$ , some basic questions we can ask about  $\sigma$  are the number of distinct tokens  $F_0(\sigma)$ , the  $k$ th frequency moment  $F_k(\sigma) = \sum_{j=1}^n f_j^k$ , the median of the collection of numbers in  $\sigma$ , and the very basic *point queries* where, given a specific  $j \in [n]$  after  $\sigma$  has been presented, we wish to know  $f_j$ . In each case, we would like an *exact* answer, not an estimate. With the trivial exception of  $F_1(\sigma)$  – which is just the length of  $\sigma$  – not one of these questions can be answered by a (possibly randomized) streaming algorithm restricted to  $o(n)$  space. However, with access to a powerful prover, things improve greatly: as shown in Chakrabarti et al. [9], point queries, median, and  $F_k$  (for integral  $k > 0$ ) can be computed exactly by a verifier using  $\tilde{O}(\sqrt{n})$  space, while receiving  $\tilde{O}(\sqrt{n})$  bits of “help” from the prover.

Notably, the protocol achieving this  $\tilde{O}(\sqrt{n})$  cost (space plus amount of help) is non-interactive: the prover sends a single message to the verifier. Chakrabarti et al. [9] also showed that under this restriction their protocol is optimal: a cost of  $\Omega(\sqrt{n})$  is required. In subsequent work, Cormode et al. [15] considered *streaming interactive proofs* (SIPs), where the verifier may engage in several rounds of interaction with the prover, seeking to minimize both the space used by the verifier and the total amount of communication. They gave SIPs with  $2k - 1$  rounds of interaction following the verifier’s single pass over the input stream, achieving a cost of  $\tilde{O}(n^{1/(k+1)})$  for the above problems. This generalizes the earlier set of results [9], which gave 1-round SIPs. Moreover, it achieves  $O(\text{polylog } n)$  cost with  $O(\log n / \log \log n)$  rounds of interaction. In recent work, Klauck and Prakash [24] further studied this kind of computation and generalized the 1-round lower bound, claiming that a  $(2k - 1)$ -round SIP must cost  $\Omega(n^{1/(k+1)})$ , even for very basic point queries.

However, we identify an implicit assumption in the Klauck–Prakash lower bound argument: it applies only to protocols in which the verifier’s messages to the prover are independent of the input. This happened to hold in all previous SIPs, which are ultimately descended from the sum-check protocol of Lund et al. [27]. Furthermore, this assumption is harmless in the classical theory of interactive proofs where public-coin protocols can simulate private-coin ones with just a polynomial blowup in cost [16]. However, these simulation results fail subtly in the streaming setting, and we show that this failure is intrinsic by giving a number of new upper bounds.

## 1.1 New Results: Exponentially Improved Constant-Round SIPs

We start by showing that even two-round SIPs are exponentially more powerful than previously believed, on certain problems. For now we state our results informally, using the  $\tilde{O}$ -notation to suppress “lower order” factors. We give formal theorem statements later in the paper, after all definitions are in place.

► **Result 1.1** (Formalized in Theorem 3.1). *There is a two-round SIP with cost  $\tilde{O}(\log n)$  for answering point queries on a stream over the universe  $[n]$ .*

The SIP that achieves this upper bound is based on an abstract protocol that we call the *polynomial evaluation protocol*. Crucially, unlike the sum-check protocols used in previous

SIPs, it involves an interaction where the verifier’s message to the prover depends on part of the input; specifically, it depends on the query. Note that two rounds of interaction is likely unavoidable in practice even if verifiability is not a concern: one round may be required for the verifier to communicate the query to the prover, with a second round required for the prover to reply.

Adding a third round of interaction allows us to answer selection queries, of which an important special case is median-finding.

► **Result 1.2** (Formalized in Theorem 3.7). *There is a three-round SIP with cost  $\tilde{O}(\log n)$  for determining the exact median of a stream of numbers from  $[n]$ .*

We can in fact answer fairly complex queries with three rounds and polylogarithmic cost. For instance, given a data set presented as a stream of points from a metric space, we can answer *exact* nearest neighbor queries to the data set very efficiently, even in high dimensions. This is somewhat surprising, given that even the offline version of the problem seems to exhibit a curse of dimensionality.

► **Result 1.3** (Formalized in Theorem 3.4). *For data sets consisting of points from  $[n]^d$  under a reasonable metric, such as the Manhattan distance  $\ell_1^d$  or the Euclidean distance  $\ell_2^d$ , there is a three-round SIP with cost  $\text{poly}(d, \log n)$  allowing exact nearest neighbor queries to the data set.*

We also give similarly efficient two-round SIPs for other well-studied query problems, such as range counting queries (Theorem 3.6), where a stream of data points is followed by a query range and the goal is to determine the number of points in the range that appeared in the stream, and pattern matching queries (Theorem 3.8), where a streamed text is followed by a (short) query pattern.

Next, we work towards a detailed understanding of the subtleties of SIPs that caused the aforementioned Klauck–Prakash lower bound [24] not to apply. Our study naturally leads into communication complexity, in particular to Arthur–Merlin communication, which we discuss next.

## 1.2 The Connection to Arthur–Merlin Communication

Like almost all previous lower bounds for data stream computations, prior SIP lower bounds [9, 24] use reductions from problems in communication complexity. To model the prover in an SIP, the appropriate setting is Arthur–Merlin communication, which we now introduce.

Suppose Alice holds an input  $x \in \mathcal{X}$ , Bob holds  $y \in \mathcal{Y}$ , and they wish to compute  $f(x, y)$  for some Boolean function  $f : \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$ , using random coins and settling for some constant probability of error. Say this costs  $R(f)$  bits of communication. Can an omniscient but untrusted Merlin, who knows  $(x, y)$ , convince “Arthur” (defined as Alice and Bob together) that  $f(x, y) = 1$ , keeping the overall communication within  $o(R(f))$ ? For several interesting functions  $f$  the answer is “Yes” and this is the general subject of Arthur–Merlin communication complexity, first considered in seminal work by Babai, Frankl, and Simon [5].

The one-pass streaming restriction on the verifier in an SIP is modeled by requiring that Alice not receive any communication from either Bob or Merlin. Thus the Alice–Bob communication is one-way, though Bob and Merlin may interact arbitrarily. We refer to this restricted communication setting as *online Arthur–Merlin communication*. It should be clear that a  $k$ -round SIP with cost  $C$  can be simulated by an online Arthur–Merlin communication

of cost  $C$  where Bob and Merlin interact for  $k$  rounds. Thus, lower bounds on SIPs would follow from corresponding communication lower bounds in the online Arthur-Merlin setting.

At this point let us recall that the classical Turing-Machine-based theory of interactive proofs considers two different models of interaction between prover and verifier, corresponding to the complexity classes  $\mathbf{IP}_{\text{TM}}$ ,<sup>1</sup> where the verifier is allowed private randomness, and  $\mathbf{AM}_{\text{TM}}$ , where he may only use public randomness. Recall the following classic results about such interactive proofs.

- **Equivalence of private and public coins.** Goldwasser and Sipser [16] proved that a  $k$ -round private coin interactive proof (à la  $\mathbf{IP}_{\text{TM}}$ ) can be simulated (with a polynomial blowup in complexity) by a  $(k + 2)$ -round public coin one (à la  $\mathbf{AM}_{\text{TM}}$ ). Thus, in the resulting protocol, the verifier can perform his interaction with the prover before even looking at the input!
- **Round reduction.** Babai and Moran [6] proved that a  $(k + 1)$ -round interactive proof can be simulated by a  $k$ -round interactive proof with a polynomial blowup in the verifier's complexity. Thus, a two-round (verifier→prover→verifier) interactive proof is just as powerful as any constant-round one.

Interestingly, as we shall show in this work, neither of these phenomena holds for the *online* communication complexity analogs of  $\mathbf{IP}_{\text{TM}}$  and  $\mathbf{AM}_{\text{TM}}$ . (Recall that “online” means that Alice does not receive any communication from either Bob or Merlin.) This point appears to have been missed in the Klauck-Prakash proof [24], which works in a “public coin” setting and thus applies only to a restricted class of SIPs. The new SIPs we design in this work correspond to a “private coin” setting, which allows the aforementioned exponential improvements.

Clearly there are nuances in online Arthur-Merlin communication complexity that do not arise in classical interactive proofs. In particular, we seek a better understanding of the precise role of rounds and of private randomness in the communication setting. This is the goal of our next batch of results.

### 1.3 New Results: Complexity Classes for Arthur-Merlin Communication

As noted above, we can think of  $\mathbf{AM}_{\text{TM}}$  as a restricted interactive proof model where the verifier must interact with the prover before looking at his input. We can then define a hierarchy of analogous communication complexity models called  $\mathbf{OMA}^{[k]}$  (Online Merlin-Arthur), where Bob interacts with Merlin in  $k$  rounds without looking at his input, and then Alice communicates with Bob one-way. We defer precise definitions to Section 4. The aforementioned Klauck-Prakash lower bound essentially says the following:

► **Proposition 1.4** (Klauck and Prakash [24]). *The INDEX problem – where Alice gets  $x \in \{0, 1\}^n$ , Bob gets  $j \in [n]$  and Bob must output  $x_j$  with high probability – requires  $\Omega(n^{1/(k+1)})$  cost in the  $\mathbf{OMA}^{[2k]}$  model.*

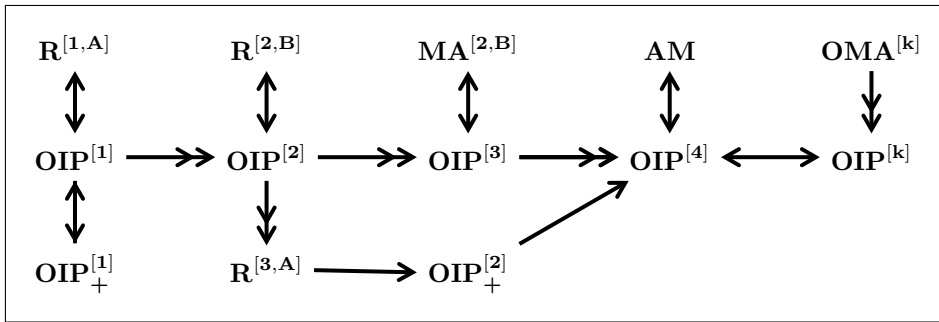
We can also define a parallel hierarchy  $\mathbf{OIP}^{[k]}$  (Online Interactive Proof) of communication analogs of  $\mathbf{IP}_{\text{TM}}$ . We now hit another subtlety. We could require the Bob-Merlin interaction to happen before the Alice→Bob communication; this is how we shall define  $\mathbf{OIP}^{[k]}$ . Alternatively, we could swap the order, so that Bob's messages to Merlin could depend on Alice's input as well; we shall call the resulting (more powerful) model  $\mathbf{OIP}_+^{[k]}$ .

<sup>1</sup> Throughout this paper, we use the subscript “TM” to denote a Turing-machine-based complexity class, to resolve the notation clash with the analogous communication complexity classes.

These communication models correspond to SIPs as follows. Every SIP designed prior to this work falls into a restricted setting where the verifier’s messages are independent of the input, so it can be simulated by an  $\mathbf{OMA}^{[k]}$  protocol with  $k$  being the number of rounds of interaction in the SIP. The SIPs we design in this work apply to “query problems” with the data set appearing before the query, and our verifier messages depend only on the query. Thus our SIPs are naturally simulable by  $\mathbf{OIP}^{[k]}$  protocols. Finally, a general SIP, where verifier messages can depend on the entire input stream, is simulable by an  $\mathbf{OIP}_+^{[k]}$  protocol.

Following Babai et al. [5], given a communication model  $\mathbf{C}$ , we define a corresponding complexity class, also denoted  $\mathbf{C}$ , consisting of all problems that have *polylogarithmic* cost protocols in the model  $\mathbf{C}$ . We now have three parallel hierarchies of communication complexity classes:  $\mathbf{OMA}^{[k]}$ ,  $\mathbf{OIP}^{[k]}$ , and  $\mathbf{OIP}_+^{[k]}$ . For our next batch of results, we prove several inclusion and separation results relating these newly defined classes to each other and to well-studied classes from earlier work in communication complexity.

► **Result 1.5** (Formalized over several theorems in Section 5). *The following complexity class inclusions and separations, given in Figure 1, hold.*



■ **Figure 1** The layout of our communication complexity zoo. An arrow from  $\mathbf{C}_1$  to  $\mathbf{C}_2$  indicates that  $\mathbf{C}_1 \subseteq \mathbf{C}_2$ . If the arrow is double-headed, then the inclusion is strict. Here  $k > 4$  is an arbitrary constant. The models  $\mathbf{R}^{[t,A]}$  (resp.  $\mathbf{R}^{[t,B]}$ ) are standard  $t$ -round randomized communication with Alice (resp. Bob) starting. The model  $\mathbf{MA}^{[2,B]}$  consists of a message from Merlin followed by Bob→Alice→Bob communication, while  $\mathbf{AM}$  is standard (see Section 5).

Notice that there are several two-way inclusions (i.e., equalities) amongst these communication complexity classes. It is worth noting that with one exception (namely  $\mathbf{OIP}^{[1]} = \mathbf{OIP}_+^{[1]}$ ) none of these equalities is trivial. For instance, consider the switch from the model  $\mathbf{R}^{[2,B]}$  to the model  $\mathbf{OIP}^{[2]}$ : Bob loses the ability to send Alice a message before hearing from her, but gains access to Merlin. It is not *a priori* clear that this switch in models will result in a complexity class that is even comparable to  $\mathbf{R}^{[2,B]}$ , and nontrivial simulation arguments (Theorems 5.3 and 5.6) are required to prove that  $\mathbf{R}^{[2,B]} = \mathbf{OIP}^{[2]}$ .

Many of our simulations incur some blowup in cost. All such blowups are at most quadratic, so polylogarithmic costs remain polylogarithmic.

The  $\mathbf{OMA}$  and  $\mathbf{OIP}$  hierarchies behave quite differently from the classical  $\mathbf{AM}_{\text{TM}}$  and  $\mathbf{IP}_{\text{TM}}$ :

- In contrast to the Goldwasser–Siper private-by-public-coin theorem, the class  $\mathbf{OIP}^{[4]}$  is strictly more powerful than  $\mathbf{OMA}^{[k]}$  (in fact, even  $\mathbf{OIP}^{[2]} \not\subseteq \mathbf{OMA}^{[k]}$ ), for every constant  $k$ .
- In contrast to the Babai–Moran round reduction theorem, there are exactly four distinct levels (not two) in the  $\mathbf{OIP}^{[k]}$  hierarchy, for constant  $k$ .

In the course of proving the separation results in Figure 1, we obtain concrete lower bounds for explicit functions that are of interest in their own right. Let us highlight one of these.

► **Result 1.6** (Formalized in Corollary 5.9). *The set disjointness problem DISJ – where Alice and Bob each get a subset of  $[n]$  and must decide whether they are disjoint – requires  $\Omega(n^{1/3})$  cost in the  $\mathbf{OIP}^{[3]}$  model and thus does not belong to the class  $\mathbf{OIP}^{[3]}$ . This lower bound is tight up to a logarithmic factor.*

This has implications for SIPs. We noted that all SIPs designed thus far (including the new ones in this work) are simulable in the weaker  $\mathbf{OIP}$  models. By a standard reduction [4] from DISJ to the frequency moments problem  $F_k$ , it follows that unlike what we achieved for point queries and median queries, *based on currently known techniques*, we cannot get a polylogarithmic cost three-round SIP for  $F_k$  ( $k \neq 1$ ).

Removing the qualifier “based on currently known techniques” above would require a similar lower bound for  $\mathbf{OIP}_+^{[3]}$ . Unfortunately, at present we are unable to prove any nontrivial lower bounds on  $\mathbf{OIP}_+^{[2]}$ , and doing so appears to be a rather difficult problem. Indeed, this inability is what led us to study the weaker  $\mathbf{OIP}$  model. Yet, because the  $\mathbf{OIP}$  models are online, the separation results in Figure 1 still morally capture the primary way in which SIPs, due to their streaming/online nature, differ from classical interactive proofs.

Finally, our result  $\mathbf{AM} = \mathbf{OIP}^{[4]}$  gives a novel characterization of  $\mathbf{AM}$  in terms of *online* communication. This is surprising because online models, where no one talks to Alice, might be expected to be too weak to capture  $\mathbf{AM}$ . Proving lower bounds on  $\mathbf{AM}$  is a longstanding and notoriously hard problem in communication complexity [26, 22, 23]. We believe our new characterization of  $\mathbf{AM}$  is of independent interest, and may prove useful in establishing non-trivial  $\mathbf{AM}$  lower bounds.

## 1.4 Related Work

### 1.4.1 Stream Computation

Early theoretical work on verifiable stream computation focused on non-interactive protocols, as in the *annotated data streams* model of Chakrabarti et al. [9]. In our language, that model corresponds to 1-round SIPs. Work in this model has established optimal protocols for several problems including frequency moments and frequent items [9]; linear algebraic problems such as matrix rank [24]; and graph problems like shortest  $s$ - $t$  path [14]. Many of these protocols have subsequently been optimized for streams whose length is much smaller than the universe size [8]. More recent protocols, such as the *Arthur–Merlin streaming protocols* of Gur and Raz [19, 8] are “barely interactive” in the sense that the prover and the verifier may exchange a constant number of messages. Meanwhile, the fully general *streaming interactive proof* (SIP) model of Cormode et al. [15, 13] permits “many” rounds of interaction. Cormode, Thaler, and Yi [15] showed that several general  $\mathbf{IP}_{\text{TM}}$  protocols can be simulated in this model. These include the powerful, general-purpose protocol of Goldwasser, Kalai, and Rothblum [18]. Given any problem in  $\mathbf{NC}_{\text{TM}}$ , the resulting protocol requires only polylogarithmic space and communication while using polylogarithmic rounds of verifier–prover interaction. Refinements and implementations of these protocols [36, 13, 35] have demonstrated scalability and the practicality of this line of work.

Algebraic techniques lie at the core of almost all nontrivial protocols in the above models. Specifically, a number of 1-round SIPs are inspired by the Aaronson–Wigderson  $\mathbf{MA}$  communication protocol for DISJ [2], which is in turn inspired by the classic sum-check



protocol of Lund et al. [27]. The sum-check protocol is also the inspiration for the way that all previous multi-round SIPs make use of interaction. The aforementioned protocol of Goldwasser et al. [18] also builds upon the sum-check protocol.

The algorithmic results outlined in Section 1.1 have a rather different algebraic idea at their core. They are based on the aforementioned *polynomial evaluation protocol*, which is obtained by adapting a result of Raz [33] about  $\mathbf{IP}/\mathbf{rpoly}_{\text{TM}}$  to a streaming setting; see the discussion at the start of Section 2.1.

Early work on interactive proofs studied space-bounded verifiers (see the survey by Condon [12]), but many protocols developed in this line of work require the verifier to store the input, and therefore do not address verifiable *stream* computation, as we do here. Goldwasser et al. [17] studied interactive proofs with verifiers in the complexity class  $\mathbf{NC}_{\text{TM}}^0$ . Interestingly, they showed that private randomness is necessary to obtain interactive proofs with verifiers in  $\mathbf{NC}_{\text{TM}}^0$ , unless the language in question is already in  $\mathbf{NC}_{\text{TM}}^0$ . This is analogous to our finding that constant-round “public coin” SIPs (where the verifier’s messages do not depend on the input) are exponentially weaker than general constant-round SIPs.

### 1.4.2 Computationally Sound Protocols

Protocols for verifiable stream computation have also been studied in the cryptography community [10, 32, 34]. These works only require soundness to hold against cheating provers that run in polynomial time. In exchange for this weaker security guarantee, these protocols can achieve properties that are impossible in the information-theoretic setting we consider. For example, they typically achieve *reusability*, allowing the verifier to use the same randomness to answer many queries. In contrast, our protocols only support “one-shot” queries, because they require the verifier to reveal secret randomness to the prover.

Chung et al. [10] combine the GKR protocol with fully homomorphic encryption (FHE) to give reusable, non-interactive protocols of polylogarithmic cost for any problem in  $\mathbf{NC}$ . Papamanthou et al. [32] give improved protocols for a class of low-complexity queries including point queries and range search: their protocols avoid the use of FHE, and allow the prover to answer such queries in polylogarithmic time (a similar property was achieved by Schröder and Schröder [34], but for a simpler class of queries, and with unidirectional communication from the verifier to the prover on each stream update). In contrast, prior work as well as our own requires the prover to spend time quasilinear in the size of the data stream after receiving a query, even if the answer itself can be computed in sublinear time (e.g., point queries, which can be solved with a single access to memory). We note however that our most interesting protocols, such as those for nearest neighbor search and pattern matching, are for problems that cannot be solved in sublinear time; hence, the quasilinear time required by our protocols does not affect the prover’s runtime by more than logarithmic factors.

### 1.4.3 Communication Complexity

Seminal work by Babai et al. [5] introduced and studied the communication analogs of the major Turing Machine complexity classes, including  $\mathbf{P}$ ,  $\mathbf{NP}$ ,  $\Sigma_2$ ,  $\Pi_2$ . They hinted at similar analogs of  $\mathbf{MA}$  and the  $\mathbf{AM}$  hierarchy. Lokam [26] related the task of placing problems outside of the communication class  $\mathbf{AM}$  to notions of matrix rigidity. He also observed that the communication complexity classes  $\mathbf{IP}$  and  $\mathbf{AM}$  behave similarly to their Turing Machine counterparts. In particular, noted theorems such as  $\mathbf{IP} = \mathbf{PSPACE}$ , Toda’s Theorem, and

Babai and Moran’s round reduction results [6] all hold in the communication world (though not under *online* communication, as shown by this work).

Online (also known as one-round) randomized communication complexity was introduced in the mid-1990s and considered by Ablayev [3], Kremer, Nisan, and Ron [25], and Newman and Szegedy [29]. Aaronson [1] introduced online variants of Merlin–Arthur communication, in classical and quantum flavors. Aaronson and Wigderson [2] gave an online **MA** communication protocol for DISJ (more generally, for INNER-PRODUCT) with cost  $\tilde{O}(\sqrt{n})$ ; this is nearly optimal, as shown by a lower bound of Klauck [22] that applies to general **MA** protocols. More recently, Klauck [23] performed a careful study of **AM**, **MA**, and its quantum analogue **QMA**. In particular, he gave a promise problem PAPPMP separating **QMA** from **AM**; we shall eventually show that PAPPMP separates **OIP**<sup>[3]</sup> from **OIP**<sup>[4]</sup>.

## 2 The SIP Model and the Polynomial Evaluation Protocol

In a data stream problem, the input  $\sigma$  is a *stream*, or sequence, of *tokens* from some data universe  $\mathcal{U}$ . The goal is to compute or approximate some function  $g(\sigma)$ , keeping space usage sublinear in the two key size parameters: (1) the length of  $\sigma$ , and (2) the size of the universe  $|\mathcal{U}|$ . Practically speaking, we would also like to process each stream update (token arrival) quickly. All our data stream algorithms will be randomized, and we shall allow them to err with some small constant probability on each input stream. In the *streaming interactive proofs* (SIP) model, after processing  $\sigma$ , the algorithm (called the “verifier”) may engage in  $k$  rounds of interaction with an oracle (the “prover”) who knows  $\sigma$  and whose goal is to lead the verifier to output the correct answer  $g(\sigma)$ . The verifier, being distrustful, will output “ $\perp$ ” (indicating “abort”) if he suspects the prover to be cheating.

All of the SIPs in this paper will work in the *turnstile streaming model*, where  $\sigma$  can contain deletions of tokens from  $\mathcal{U}$ , in addition to insertions. In this model it is best to think of the input as being a stream of integer *updates* to a vector  $\mathbf{x} = (x_1, \dots, x_n) \in \mathbb{Z}^n$ . Initially  $\mathbf{x} = \mathbf{0}$ , and an update is a tuple  $(i, c) \in [n] \times \mathbb{Z}$ , which has the effect of adding  $c$  to the entry  $x_i$ . We will sometimes describe our algorithms as they apply to the vanilla streaming model, but it will be straightforward to extend them to the turnstile model.

We say that an SIP computes the function  $g$  with *completeness error*  $\varepsilon_c$  and *soundness error*  $\varepsilon_s$  if for all inputs  $\mathbf{x}$  there exists a prover strategy that will cause the verifier to output  $g(\mathbf{x})$  with probability at least  $1 - \varepsilon_c$ , and no prover strategy can cause the verifier to output a value outside  $\{g(x), \perp\}$  with probability larger than  $\varepsilon_s$ . In designing SIPs, our goal will be to achieve  $\varepsilon_c, \varepsilon_s \leq 1/3$ ; clearly the theory remains unchanged if we replace  $1/3$  by another constant in  $(0, 1/2)$ . A SIP with  $\varepsilon_c = 0$  is said to have *perfect completeness*. The total length of the verifier–prover interaction is the *help cost*. The space used by the streaming verifier is the *space cost*. The *cost* of an SIP is the sum of its help cost and its space cost. When designing SIP protocols we will also discuss the time complexities of the prover and the verifier. To keep things simple, we consider a model in which all arithmetic operations on a finite field of size  $n^{O(1)}$  can be executed in unit time.

### 2.1 The Polynomial Evaluation Protocol

We shall present a two-round SIP for an abstract data stream problem called “polynomial evaluation,” where the input consists of a multivariate polynomial described implicitly, as a table of values, followed by a point at which the polynomial must be evaluated. Without space constraints, this problem simply amounts to interpolation followed by direct evaluation, but our goal is to obtain a protocol where the verifier uses space roughly logarithmic in the

size of the table of values, and is convinced by the prover about the correct answer after a similar amount of communication. For ease of presentation, we shall first consider a concrete special setting that is important in its own right: the INDEX problem. In this problem, the input is a stream of  $n$  data bits  $x_1, \dots, x_n$ , followed by a query index  $j \in [n]$ . The goal is to output  $x_j$  with error at most  $1/3$ .

With very different motivations from ours, Raz [33] gave an interactive proof protocol placing every language in  $\mathbf{IP}_{\text{TM}}/\mathbf{rpolynomial}$ , the class of languages that have interactive proofs with polynomial-time verifiers that take randomized advice, where the advice is kept secret from the prover. Our SIP for INDEX can be seen as an adaptation of Raz’s interactive proof to the streaming setting.

► **Theorem 2.1.** *The INDEX problem has a two-round SIP with cost  $O(\log n \log \log n)$ , in which the verifier processes each stream token in  $O(\log n)$  time and the prover runs in total time  $O(n \log n)$ .*

**Proof.** Assume WLOG that  $n = 2^b$ , for some integer  $b$ . Identify each integer  $z \in [n]$  with a Boolean vector  $\mathbf{z} = (z_1, \dots, z_b) \in \{0, 1\}^b$  in some canonical way, such as by using the binary representation of  $z$ . We can then view the data bits as a table of values for the Boolean function  $g_x : \{0, 1\}^b \rightarrow \{0, 1\}$  given by  $g_x(\mathbf{z}) = x_z$ , and thus for the multilinear  $b$ -variate polynomial  $\tilde{g}_x(Z_1, \dots, Z_b)$  given by

$$\tilde{g}_x(Z_1, \dots, Z_b) = \sum_{\mathbf{z} \in \{0, 1\}^b} g_x(\mathbf{z}) \chi_{\mathbf{z}}(Z_1, \dots, Z_b), \text{ where} \tag{1}$$

$$\chi_{\mathbf{u}}(Z_1, \dots, Z_b) = \prod_{i=1}^b ((1 - u_i)(1 - Z_i) + u_i Z_i) \tag{2}$$

is the indicator function of the vector  $\mathbf{u} = (u_1, \dots, u_b)$ . We shall interpret  $\tilde{g}_x$  as a polynomial in  $\mathbb{F}[Z_1, \dots, Z_b]$  for a fixed “large enough” finite field  $\mathbb{F}$ . With this interpretation,  $\tilde{g}_x$  is called the multilinear extension of  $g_x$  to  $\mathbb{F}$ . We define a *line* in  $\mathbb{F}^b$  to be the range of a nonconstant affine function from  $\mathbb{F}$  to  $\mathbb{F}^b$ . Every line contains exactly  $|\mathbb{F}|$  points. Given such a line,  $\ell$ , we define its *canonical representation* to be the degree-1 polynomial  $\lambda_{\ell}(W) \in \mathbb{F}^b[W]$  such that  $\lambda_{\ell}(0)$  and  $\lambda_{\ell}(1)$  are, respectively, the lexicographically first and second points in  $\ell$ . We define the *canonical restriction* of a polynomial  $f(Z_1, \dots, Z_b)$  to  $\ell$  to be the univariate polynomial  $f(\lambda_{\ell}(W)) \in \mathbb{F}[W]$ , whose degree is at most the total degree of  $f$ .

Using the above notations and conventions, our two-round SIP for INDEX works as shown in Figure 2.

To analyze this protocol, first note that after reading all the data bits, the verifier would have computed  $Q = \tilde{g}_x(\mathbf{r})$ , by Eq. (1). Now the protocol is easily seen to have perfect completeness. Since  $\tilde{g}_x(Z_1, \dots, Z_b)$  is multilinear, it follows that  $\deg(\tilde{g}_x(\lambda_{\ell}(W))) \leq b$ , so the prover can always honestly choose  $h(W) = \tilde{g}_x(\lambda_{\ell}(W))$ . If he does so, then we will indeed have  $h(t) = \tilde{g}_x(\lambda_{\ell}(t)) = \tilde{g}_x(\mathbf{r}) = Q$ , and the verifier’s check will pass. Finally, the verifier will output  $h(w) = \tilde{g}_x(\lambda_{\ell}(w)) = \tilde{g}_x(\mathbf{j}) = x_j$ , the correct answer to the INDEX instance.

Next, we analyze soundness. If the prover supplies a polynomial  $h(W) \neq \tilde{g}_x(\lambda_{\ell}(W))$ , then, since both polynomials have degree at most  $b$ , they agree at at most  $b$  points in  $\mathbb{F}$ . From the prover’s perspective after he receives the verifier’s message,  $\mathbf{r}$  is uniformly distributed in  $\ell \setminus \{\mathbf{j}\}$ . Thus,  $\Pr_{\mathbf{r}}[h(t) = Q] \leq b/(|\mathbb{F}| - 1) \leq 1/3$ .

Now we consider this protocol’s costs. The verifier maintains the random point  $\mathbf{r} \in \mathbb{F}^b$  and the running sum  $Q \in \mathbb{F}$ , using  $O(b \log |\mathbb{F}|)$  space. He sends the prover  $\ell$ , which is specified by two elements of  $\mathbb{F}^b$ , and receives a degree- $b$  polynomial in  $\mathbb{F}[W]$ ; both communications use at most  $O(b \log |\mathbb{F}|)$  bits. Recalling that  $|\mathbb{F}| \leq 6b + 2$ , we see that both space and communication costs are in  $O(b \log b) = O(\log n \log \log n)$ .

<p><b>Input:</b> Stream of data bits <math>(x_1, \dots, x_n)</math> where <math>n = 2^b</math>, followed by index <math>j \in [n]</math>.</p> <p><b>Goal:</b> Prover to convince Verifier to output the correct value of <math>x_j</math>.</p> <p><b>Shared Agreement:</b> Finite field <math>\mathbb{F}</math> with <math>3b + 1 \leq  \mathbb{F}  \leq 6b + 2</math>; bijective map <math>u \in [n] \longleftrightarrow \mathbf{u} \in \{0, 1\}^b</math>.</p> <hr/> <p><b>Initialization:</b> Verifier picks <math>\mathbf{r} \in_R \mathbb{F}^b</math> uniformly at random, sets <math>Q \leftarrow 0</math>.</p> <p><b>Stream Processing:</b> Upon reading <math>x_z</math>, where <math>z \in [n]</math>, Verifier updates <math>Q \leftarrow Q + x_z \chi_{\mathbf{z}}(\mathbf{r})</math>.</p> <p><b>Query Handling:</b> Upon reading the index <math>j</math>, Verifier interacts with Prover as follows:</p> <ol style="list-style-type: none"> <li>1. If <math>\mathbf{j} = \mathbf{r}</math>, Verifier outputs <math>Q</math> as the answer. Otherwise, he sends Prover <math>\ell</math>, the unique line in <math>\mathbb{F}^b</math> through <math>\mathbf{j}</math> and <math>\mathbf{r}</math>.</li> <li>2. Prover sends Verifier a polynomial <math>h(W) \in \mathbb{F}[W]</math> of degree at most <math>b</math>, claiming that it is the canonical restriction of the multilinear polynomial <math>\tilde{g}_x(Z_1, \dots, Z_b)</math> to the line <math>\ell</math>. That is, Prover claims that <math>h(W) \equiv \tilde{g}_x(\lambda_\ell(W))</math>.</li> <li>3. Let <math>w, t \in \mathbb{F}</math> be such that <math>\lambda_\ell(w) = \mathbf{j}</math> and <math>\lambda_\ell(t) = \mathbf{r}</math>. Verifier checks that <math>h(t) = Q</math>, aborting if not. If the check passes, Verifier outputs <math>h(w)</math> as the answer.</li> </ol>
---

■ **Figure 2** A Two-Round Streaming Interactive Proof (SIP) Protocol for the INDEX Problem

Finally, we consider the verifier's and prover's runtimes. The honest prover must send the univariate polynomial  $\tilde{g}_x(\lambda_\ell(W))$ . Since  $\tilde{g}_x$  has degree at most  $b$ , it suffices for the prover to specify the evaluations of  $\tilde{g}_x(\lambda_\ell(W))$  at  $b + 1 = O(\log n)$  points. A direct application of Qqs. (1) and (2) shows that each evaluation can be done in  $O(n \log n)$  time, resulting in a total runtime of  $O(n \log^2 n)$ . However, using now-standard memoization techniques (see e.g. [36, Section 5.1]), it is possible for the prover to in fact perform each of these evaluations in just  $O(n)$  time, resulting in a total runtime of  $O(n \log n)$ . The verifier can run in  $O(b) = O(\log n)$  time per stream update, as each stream update  $x_z$  only requires the verifier to compute  $\chi_{\mathbf{z}}(\mathbf{r})$ , and it follows from Eq. (2) that this can be done with  $O(b)$  field operations. When interacting with the prover, the verifier first needs to determine the line  $\ell$  through  $\mathbf{j}$  and  $\mathbf{r}$ , which he can do in  $O(b) = O(\log n)$  time. To process the prover's reply, he must evaluate the polynomial  $h$  at the points  $t$  and  $w$ ; these evaluations can be done in  $\text{polylog } n$  time. ◀

The above SIP protocol uses very little of the special structure of the INDEX problem. Let us abstract out its salient features, so as to handle the general problem described at the start of this section. First, note the protocol treats the data set given by  $(x_1, \dots, x_n)$  as an implicit description of the polynomial  $\tilde{g}_x$ . Second, note that our soundness analysis did not require multilinearity per se, only an upper bound on the total degree of  $\tilde{g}_x$ . Finally, note that the specific form of Eqs. (1) and (2) is not crucial either; all we used was that it allows the verifier an easy streaming computation. Thus, we obtain the following generic result.

► **Theorem 2.2** (Polynomial Evaluation Protocol). *Suppose an input data stream implicitly describes a  $v$ -variate polynomial  $g$  of total degree  $d$  over a field  $\mathbb{F}$ , followed by a point  $\mathbf{j} \in \mathbb{F}^v$ . Suppose this implicit description allows a streaming verifier to evaluate  $g$  at a random point  $\mathbf{r} \in_R \mathbb{F}^v$  using space  $S$ . Then the technique of the protocol in Figure 2 gives a two-round SIP for computing  $g(\mathbf{j})$ , with the following properties: (1) perfect completeness; (2) soundness error bounded by  $d/(|\mathbb{F}| - 1)$ ; (3) space usage in  $O(v \log |\mathbb{F}| + S)$ ; (4) help cost in  $O((d + v) \log |\mathbb{F}|)$ . ◀*

We shall refer to the abstract protocol given by Theorem 2.2 as the polynomial evaluation protocol.

### 3 Constant-Round SIPs for Query Problems

We shall now apply the polynomial evaluation protocol to design SIPs proving the various upper bounds outlined in Section 1.1. The first application is immediate; later applications bring in additional ideas.

#### 3.1 Point Queries.

In the POINTQUERY problem, the input is a stream in the turnstile model, updating an initially-zero vector  $\mathbf{x} \in \mathbb{Z}^n$ , followed by a query  $j \in [n]$ . The goal is to output  $x_j$ .

► **Theorem 3.1.** *Suppose the input to POINTQUERY is guaranteed to satisfy  $|x_i| \leq q$  at end of the data stream, for all entries of  $\mathbf{x}$ , where the bound  $q$  is known a priori. Then there is a two-round SIP for POINTQUERY with space and help costs in  $O(\log n \log(q + \log n))$ .*

**Proof.** Assume WLOG that  $n = 2^b$  for an integer  $b$ , and use a bijection  $u \in [n] \longleftrightarrow \mathbf{u} \in \{0, 1\}^b$  as in Theorem 2.1. The vector  $\mathbf{x}$  resulting from the updates defines a multilinear polynomial  $\tilde{g}_{\mathbf{x}}(Z_1, \dots, Z_b)$  by Eq. (1), where  $g_{\mathbf{x}}(\mathbf{z}) := x_{\mathbf{z}}$ . We can treat  $\tilde{g}_{\mathbf{x}}$  as a polynomial over any field we like, but to solve our problem, we need to tell apart the  $2q + 1$  possible values taken on by the entries of  $\mathbf{x}$  (recall that  $q$  is an upper bound on  $\|\mathbf{x}\|_{\infty}$  at the end of the stream). For this it suffices to have  $\text{char}(\mathbb{F}) \geq 2q + 1$ .

Applying the polynomial evaluation protocol is now straightforward. The verifier starts with  $\mathbf{r} \in_R \mathbb{F}^b$  and  $Q = 0$ . Upon receiving an update indicating “ $x_i \leftarrow x_i + c$ ,” he updates  $Q \leftarrow Q + c\chi_i(\mathbf{r})$ . The other details are as in Figure 2. The space and communication costs are both in  $O(b \log |\mathbb{F}|)$  as before.

To ensure a soundness error of at most  $1/3$ , we let  $|\mathbb{F}| > 3b$  as before. This and the earlier condition on  $\text{char}(\mathbb{F})$  can both be satisfied by, e.g., taking  $\mathbb{F} = \mathbb{F}_p$ , for a prime  $p > 3b + 2q$ . This translates to cost bounds in  $O(\log n \log(q + \log n))$ , as claimed. ◀

#### 3.2 Nearest Neighbor Queries

Consider a “premetric” space<sup>2</sup>  $(\mathcal{X}, D)$  given by a finite ground set  $\mathcal{X}$  and distance function  $D : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}^+$  satisfying  $D(\mathbf{x}, \mathbf{x}) = 0$  for all  $\mathbf{x} \in \mathcal{X}$ . Let  $B_D(\mathbf{z}, r) = \{\mathbf{x} \in \mathcal{X} : D(\mathbf{x}, \mathbf{z}) \leq r\}$  denote the corresponding ball of radius  $r \in \mathbb{R}^+$  centered at  $\mathbf{z} \in \mathcal{X}$ . In the NEARESTNEIGHBOR problem, the input consists of a stream  $\langle \mathbf{x}^{(1)}, \dots, \mathbf{x}^{(m)} \rangle$  of  $m$  points from  $\mathcal{X}$ , constituting the data set, followed by a query point  $\mathbf{z} \in \mathcal{X}$ . The goal is to output  $\mathbf{x}^* = \arg \min_{\mathbf{x}^{(i)}} D(\mathbf{x}^{(i)}, \mathbf{z})$ , the nearest neighbor of  $\mathbf{z}$  in the data set. We shall give highly efficient SIPs for this problem that handle rather general distance functions  $D$ . To keep our statements of bounds simple, we shall impose the following structure on  $(\mathcal{X}, D)$ .

- We assume that  $\mathcal{X} = [n]^d$ . We think of  $d$  as the dimensionality of the data, and  $[n]^d$  as a very fine “grid” over the ambient space of possible points.
- For all  $\mathbf{x}, \mathbf{y} \in [n]^d$ ,  $D(\mathbf{x}, \mathbf{y}) \leq 1$  is an integer multiple of a small parameter  $\varepsilon \geq 1/n^d$ .

Overall, this amounts to assuming that our data set has polynomial *spread*: the ratio between the maximum and minimum distance. We proceed to give two SIPs for NEARESTNEIGHBOR. Our basic SIP has cost roughly logarithmic in the stream length and the spread (and therefore

<sup>2</sup> This very general setting, which includes metric spaces as special cases, captures several important distance functions such as the Bregman divergences from information theory and machine learning that satisfy neither symmetry nor the triangle inequality.

linear in  $d$  but only logarithmic in  $n$ ). After we present it, we shall critique it and then give a more sophisticated SIP to handle its faults.

► **Theorem 3.2.** *Under the above assumptions on the premetric space  $(\mathcal{X}, D)$ , the NEARESTNEIGHBOR problem has a three-round SIP with cost  $O(d \log n \log(m + \log(d \log n)))$ .*

**Proof.** Let  $\mathcal{B} = \{B_D(\mathbf{x}, j\varepsilon) : \mathbf{x} \in \mathcal{X}, j \in \mathbb{Z}, 0 \leq j \leq 1/\varepsilon\}$  be the set of all balls of all radii between 0 and 1 (quantized at granularity  $\varepsilon$ ). By our assumptions on the structure of  $(\mathcal{X}, D)$ , we have  $|\mathcal{B}| \leq n^d/\varepsilon \leq n^{2d}$ . The input stream  $\langle \mathbf{x}^{(1)}, \dots, \mathbf{x}^{(m)} \rangle$  defines a *derived stream*, consisting of updates to a vector  $\mathbf{v}$  indexed by the elements of  $\mathcal{B}$ . We shall denote by  $v[\beta]$  the entry of  $\mathbf{v}$  indexed by  $\beta \in \mathcal{B}$ . The derived stream is defined as follows: the token  $\mathbf{x}^{(i)}$  increments  $v[\beta]$  for every ball  $\beta$  that contains  $\mathbf{x}^{(i)}$ . The verifier runs the POINTQUERY protocol of Theorem 3.1 on this derived stream.

The verifier learns the query point  $\mathbf{z}$  at the end of the stream. The prover then supplies a point  $\mathbf{y}$  claimed to be a valid nearest neighbor (note that there may be more than one valid answer). To check this claim, it is sufficient for the verifier to check two properties: (1) that  $\mathbf{y}$  did appear in the stream, and (2) that the stream contained no point closer to  $\mathbf{z}$  than  $\mathbf{y}$ . The first property holds iff  $v[B_D(\mathbf{y}, 0)] \neq 0$ . The second property holds iff  $v[B_D(\mathbf{z}, D(\mathbf{y}, \mathbf{z}) - \varepsilon)] = 0$ . Clearly, these two properties can be checked by two point queries over the derived stream.

Following the protocol of Theorem 3.1, the two point queries (executed in parallel) involve two more rounds between the verifier and the prover, for an overall three-round SIP. Since the entries of  $\mathbf{v}$  never exceed  $m$ , each POINTQUERY protocol requires space and help costs  $O(d \log n \log(m + \log(d \log n)))$ . ◀

While the protocol of Theorem 3.2 achieves very small space and help costs, the prover's and verifier's runtimes could be as high as  $\Omega(n^d)$ , because processing a single stream token  $\mathbf{x}^{(i)}$  may require both parties to enumerate all balls containing  $\mathbf{x}^{(i)}$ . Ultimately, this inefficiency is because the protocol assumes hardly anything about the nature of the distance function  $D$  and, as a result, does not get to exploit any structural information about the balls in  $\mathcal{B}$ .

To rectify this, we shall make the entirely reasonable assumption that the distance function  $D$  is “efficiently computable” in the rather mild sense that membership in a ball generated by  $D$  can be decided by a short (say, polynomial-length) formula. Accordingly, we shall express our bounds in terms of a parameter that captures this notion of efficient computation.

► **Definition 3.3.** Suppose the distance function  $D$  on  $\mathcal{X}$  satisfies the assumptions for Theorem 3.2. Let  $\Phi_D : \mathcal{B} \times \mathcal{X} \rightarrow \{0, 1\}$  be the ball membership function for  $D$ , i.e.,  $\Phi_D(B_D(\mathbf{z}, r), \mathbf{x}) = 1 \iff \mathbf{x} \in B_D(\mathbf{z}, r)$ . Think of  $\Phi_D$  as a Boolean function of  $(3d \log n)$ -bit inputs. We define the *formula size complexity* of  $D$ , denoted  $\text{fsize}(D)$ , to be the length of the shortest de Morgan formula for  $\Phi_D$ .

Since addition and multiplication of  $b$ -bit integers can both be computed by Boolean circuits in depth  $\log b$  (see, e.g., [31, 37]), they can be computed by Boolean formulae of size  $\text{poly}(b)$ . It follows that for many natural distance functions  $D$ , including the Euclidean, Hamming,  $\ell_1$ , and  $\ell_\infty$  metrics (and in fact  $\ell_p$  for all suitably “small” positive  $p$ ), we have  $\text{fsize}(D) = \text{poly}(d, \log n)$ .

► **Theorem 3.4.** *Suppose the premetric space  $(\mathcal{X}, D)$  satisfies the assumptions made for Theorem 3.2. Then NEARESTNEIGHBOR on  $(\mathcal{X}, D)$  has a three-round SIP, whose space and help costs are both at most  $O(\text{fsize}(D) \log(m + \text{fsize}(D)))$ , in which the verifier processes each*



stream update in time  $O(\text{fsize}(D))$ , and the prover runs in total time  $m \cdot \text{poly}(\text{fsize}(D))$ . In particular, if  $\text{fsize}(D) = \text{poly}(d, \log n)$ , as is the case for many natural distance functions  $D$ , then the space and help costs are both  $\text{poly}(d, \log m, \log n)$ , the verifier runs in time  $\text{poly}(d, \log n)$  per stream update, and the prover runs in total time  $m \cdot \text{poly}(d, \log n)$ .

We defer a proof of Theorem 3.4 to the full version of the paper, but the high level idea that allows us to avoid the high runtimes of the previous protocol is as follows. Essentially, the SIP of Theorem 3.2 ran our polynomial evaluation protocol on a *multilinear* extension of the vector  $\mathbf{v}$  defined by the derived stream. That SIP took  $\mathbf{v}$  to be a completely arbitrary table of values. As a result, the verifier’s computation – evaluating the multilinear extension at a random point – became costly. The honest prover incurred similar costs. A closer examination of the nature of  $\mathbf{v}$  reveals that if  $D$  is a “reasonable” distance function, then  $\mathbf{v}$  itself has plenty of structure. In particular, an appropriate *higher degree* extension of  $\mathbf{v}$  can in fact be evaluated much more efficiently (by both the verifier and the prover) than the above multilinear extension.

### 3.3 Range Counting Queries

Let  $\mathcal{U}$  be any data universe and  $\mathcal{R} \subseteq 2^{\mathcal{U}}$  a set of *ranges*. In the RANGECOUNT problem, the data stream  $\sigma = \langle \mathbf{x}^{(1)}, \dots, \mathbf{x}^{(m)}, R^* \rangle$  specifies a sequence of universe elements  $\mathbf{x}^{(i)} \in \mathcal{U}$ , followed by a *query* or *target* range  $R^* \in \mathcal{R}$ . The goal is to output  $|\{i : \mathbf{x}^{(i)} \in R^*\}|$ , i.e., the number of elements in the target range that appeared in the stream.

We easily obtain a *two-round* streaming interactive proof for the RANGECOUNT problem with cost bounded by  $O(\log |\mathcal{R}| \log(|\mathcal{R}|m))$ . The verifier simply runs a POINTQUERY on the derived stream  $\sigma'$  defined to have data universe  $\mathcal{R}$ .  $\sigma'$  is obtained from  $\sigma$  as follows: on each stream update  $\mathbf{x}^{(i)} \in \mathcal{U}$ , the verifier inserts into  $\sigma'$  one copy of each range  $R \in \mathcal{R}$  such that  $\mathbf{x}^{(i)} \in R$ . The range count problem is equivalent to a POINTQUERY on  $\sigma'$ , with the target item being  $R^*$ , and we obtain the following theorem.

► **Theorem 3.5.** *There is a two-round SIP with  $O(\log |\mathcal{R}| \log(|\mathcal{R}|m))$  cost for RANGECOUNT.*

In particular, for spaces of bounded *shatter dimension*  $\rho$ ,  $\log |\mathcal{R}| = \rho \log m = O(\log m)$ . The above protocol also implies a *three-round* SIP for the problem of *linear classification*, a core problem in machine learning. Just like the protocol for NEARESTNEIGHBOR invokes a two-round protocol for INDEX, an SIP for linear classification (find a hyperplane that separates red and blue points) verifies that the proposed hyperplane is empty of red points on one side and blue points on the other using the above two-round RANGECOUNT protocol.

The prover and verifier in the protocol of Theorem 3.5 may require time  $\Omega(|\mathcal{R}|)$  per stream update. This could be prohibitively large. However, we can obtain savings analogous to Theorem 3.4 if we make a mild “efficient computability” assumption on our ranges. Specifically, suppose there exists a (poly( $S$ )-time uniform) de Morgan formula  $\Phi$  of length  $S$  that takes as input a binary string representing a point  $\mathbf{x}^{(i)} \in \mathcal{U}$ , as well as the label of a range  $R \in \mathcal{R}$  and outputs a bit that is 1 if and only if  $\mathbf{x}^{(i)} \in R$ . We then obtain the following more practical SIP.

► **Theorem 3.6.** *Suppose membership in ranges from  $\mathcal{R}$  can be decided by de Morgan formulas of length  $S$  as above. Then there is a two-round SIP for RANGECOUNT on  $\mathcal{R}$ , with costs at most  $O(S \log(m + S))$ , in which the verifier runs in time  $O(S)$  per stream update, and the prover runs in total time  $m \cdot \text{poly}(S)$ .*

### 3.4 Median and Selection Queries

We give a three-round SIP for SELECTION, of which MEDIAN is a special case. In the SELECTION problem, defined over data universe  $\mathcal{U} = [n]$ , the data stream  $\sigma = \langle \mathbf{x}^{(1)}, \dots, \mathbf{x}^{(m)}, \rho \rangle$  is a sequence of elements from  $[n]$ , followed by a desired rank  $\rho \in [m]$ . For  $i \in [n]$ , let  $f_i := \{j : \mathbf{x}^{(j)} = i\}$  denote the number of times element  $i$  appears in the stream. Given a desired rank  $\rho \in [m]$ , the goal is to output an element  $j \in [n]$  such that

$$\sum_{k < j} f_k < \rho \quad \text{and} \quad \sum_{k > j} f_k \leq m - \rho. \quad (3)$$

MEDIAN is the special case of SELECTION when  $\rho = \lfloor m/2 \rfloor$ .

Our three-round SIPs for SELECTION essentially work by reducing to the RANGECOUNT problem, but an extra round is required for the prover to send the desired element  $j$  to the verifier.

► **Theorem 3.7.** *There is a three-round SIP for SELECTION with cost at most  $O(\log n \log(m + \log n))$  in which the verifier runs in time  $\text{poly}(\log n, \log m)$  per update, and the prover runs in total time  $m \cdot \text{poly}(\log n, \log m)$ .*

The proof of Theorem 3.7 is deferred to the full version of the paper.

### 3.5 Pattern Matching Queries

In the pattern matching with wildcards problem, denoted PMW, we are given a stream  $\sigma$  representing text  $T = (t_1, \dots, t_m) \in \{0, 1, *\}^m$  followed by a pattern  $P = (p_1, \dots, p_q) \in \{0, 1, *\}^q$ . The wildcard symbol  $*$  is interpreted as “don’t care”, and the pattern  $P$  is said to occur at location  $i$  in  $T$  if, for every position  $j$  in  $P$ , either  $p_j = t_{i+j}$  or at least one of  $p_j$  and  $t_{i+j}$  is the wildcard symbol. The PMW problem is to determine the number of locations at which  $P$  occurred in  $T$ . PATTERNMATCHING refers to the special case where “don’t care” symbols are not permitted. We focus on a binary alphabet; a larger alphabet  $\mathcal{U}$  can be handled by replacing each character in  $\mathcal{U}$  with its binary representation, growing the parameter  $q$  by a factor of  $\log |\mathcal{U}|$ .

Pattern matching, both with and without wildcards, has been extensively studied within the algorithmic literature, with applications ranging from internet search to computational genetics (see e.g. [11, 20] and the references therein). Verifiable protocols for pattern matching enable searching in the cloud, and complements work on searching in encrypted data within the cloud (e.g. [7]). Cormode et al. [13] described and implemented an SIP for PMW that required roughly  $\Theta(\log^2 m)$  rounds and had space help costs bounded by  $\tilde{O}(\log^2 m)$ ; Concretely, their implementation required well over 1,000 rounds, even for quite small streams (of length  $2^{17}$ ). In stark contrast, our new protocol requires the optimal number of rounds: two.

► **Theorem 3.8.** *There is a 2-round SIP for PMW with space and help costs at most  $O(q \log(q + m))$ , in which the verifier runs in time  $O(q)$  per stream update, and the prover runs in total time  $m \cdot \text{poly}(q)$ .*

The proof of Theorem 3.8 is deferred to the full version of the paper. We remark that the PMW protocol of Theorem 3.8 can be run even if the verifier only knows an upper bound on the length  $q$  of the pattern. This is because, for any  $q' \leq q$ , a pattern  $P' \in \{0, 1, *\}^{q'}$  is equivalent to the pattern  $P \in \{0, 1, *\}^q$  obtained from  $P'$  by concatenating  $q - q'$  wildcard symbols to  $P'$ .



## 4 Communication Protocols and Complexity Classes

We now turn to the study of communication complexity classes motivated by a desire to understand streaming interactive proofs (SIPs) from a complexity-theoretic viewpoint. In this section, we lay out the necessary definitions and terminology to rigorously discuss the notions outlined in Section 1.3. In the next section we prove the many parts of Result 1.5.

### 4.1 Definitions

Communication problems arise naturally out of data stream problems if we suppose Alice holds a prefix of the input stream, and Bob the remaining suffix. The primary goal of such reductions is to obtain space lower bounds on data stream algorithms, so we are free to split the stream at any place we like. For example, most data stream problems in Section 3 are *query problems*, where the input consists of a streamed data set,  $S$ , followed by a query,  $q$ , to apply to  $S$ . In this case, it would be natural to split the input by giving  $S$  to Alice and  $q$  to Bob. Communication problems that will play an important role in this paper include the index problem  $\text{INDEX} : \{0, 1\}^n \times [n] \rightarrow \{0, 1\}$  where  $[n] := \{1, \dots, n\}$  and  $\text{INDEX}(x, j) = x_j$ , the set-intersection and set-disjointness problems  $\text{INTER}, \text{DISJ} : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$  where  $\text{INTER}(x, y) = \neg \text{DISJ}(x, y) = \bigvee_{i=1}^n (x_i \wedge y_i)$ , and the median relation  $\text{MED} : [n]^m \times [n]^m \rightarrow [n]$ , where inputs  $x, y \in [n]^m \times [n]^m$  are interpreted as two halves of a list of numbers, and the valid output(s) corresponds to the median(s) of the combined list.

#### 4.1.1 Communication Complexity Classes

All our communication models provide random coins and allow two-sided error probability up to a constant; when unspecified, this constant defaults to  $1/3$ . Given a communication model  $\mathbf{C}$ , we denote the corresponding complexity measure of a problem  $f$  by  $C(f)$ . Following Babai et al. [5], we also denote by  $\mathbf{C}$  the corresponding complexity class, defined as the set of all functions  $f : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$  such that  $C(f) = (\log n)^{O(1)}$ , i.e., functions that are “easy” in the model  $\mathbf{C}$ .

We let  $\mathbf{R}^{[k, \mathbf{A}]}$  denote the model of randomized communication complexity where Alice and Bob exchange  $k \geq 1$  messages in total with Alice sending the first;  $\mathbf{R}^{[k, \mathbf{B}]}$  is similar, except that Bob starts. In the  $\mathbf{MA}$  model, the super-player Merlin, who sees all of the input, broadcasts a message at the start, following which Alice and Bob run a (two-way, arbitrary-round) randomized “verification” protocol. The  $\mathbf{MA}^{[k, \mathbf{A}]}$  and  $\mathbf{MA}^{[k, \mathbf{B}]}$  models are restrictions of  $\mathbf{MA}$  where Merlin speaks only to Bob<sup>3</sup> and the verification protocol following Merlin’s single message is restricted to lie in  $\mathbf{R}^{[k, \mathbf{A}]}$  and  $\mathbf{R}^{[k, \mathbf{B}]}$  respectively.

The  $\mathbf{MA}$  model (indeed, its restriction  $\mathbf{MA}^{[1, \mathbf{A}]}$ ) allows us to simulate 1-round SIPs in an obvious way: Merlin sends Bob the prover’s message, and Alice sends Bob the verifier’s memory contents after it has processed her prefix of the stream. Notice that the order of the two messages is not important, modulo one crucial consideration: Alice must have a private channel to Bob and the random coins used to generate the message from Alice to Bob must be *hidden coins*, invisible to Merlin but shared between Alice and Bob (which is why we called them “hidden coins” rather than “private coins”).

The models  $\mathbf{OMA}^{[k]}$ ,  $\mathbf{OIP}^{[k]}$ , and  $\mathbf{OIP}_+^{[k]}$ , for  $k \geq 1$ , are obtained by extending  $\mathbf{MA}^{[1, \mathbf{A}]}$  to simulate  $k$ -round SIP protocols. These communication models work as follows. In each

<sup>3</sup> Our definition breaks symmetry between Alice and Bob because our eventual goal is to study online protocols.

case, Alice and Bob first toss some hidden coins. Then, upon receiving the input, two things happen: (1) Merlin and Bob interact for  $k$  rounds, with Merlin sending the last message in the interaction, and (2) Alice sends Bob a message, randomized using the hidden coins. After these actions are completed, Bob produces an output in  $\{0, 1\}$ . The differences between the three series of models are as follows.

- In  $\mathbf{OMA}^{[k]}$ , (1) happens before (2) and Bob must interact with Merlin before looking at his input. This is directly analogous to  $\mathbf{AM}_{\text{TM}}$ ; see the discussion in Section 1.2.
- In  $\mathbf{OIP}^{[k]}$ , (1) happens before (2) and Bob may look at his input before talking to Merlin.
- Finally,  $\mathbf{OIP}_+^{[k]}$  is like  $\mathbf{OIP}^{[k]}$  except that (2) happens before (1). Thus, Bob’s messages may depend on Alice’s actual message to Bob, not just on Bob’s input and the hidden coins.

In the  $\mathbf{AM}$  model, the parties first choose a public random string, then Merlin broadcasts a message to Alice and Bob, who then run a deterministic communication protocol to arrive at a Boolean output. Since Merlin can in fact predict the exact transcript that Alice and Bob will generate following his message, we can assume without loss of generality that after Merlin’s message, Alice and Bob output one bit each indicating whether or not they accept Merlin’s prediction.

#### 4.1.2 Cost and Value of Protocols

Let  $\mathcal{P}$  be a protocol in a model  $\mathbf{C}$  involving Merlin. For each input  $(x, y)$ ,  $\mathcal{P}$  defines a game between Merlin and Arthur (recall that Alice and Bob together constitute Arthur), wherein Merlin’s goal is to make Arthur output 1. We define the *value*  $V^{\mathcal{P}}(x, y)$  to be Merlin’s probability of winning this game with optimal play. Given a Boolean function  $f$ , we say that  $\mathcal{P}$  computes  $f$  with *soundness error*  $\varepsilon_s$  and *completeness error*  $\varepsilon_c$  if, for all  $x, y$  we have

$$f(x, y) = 0 \Rightarrow V^{\mathcal{P}}(x, y) \leq \varepsilon_s, \quad \text{and} \quad f(x, y) = 1 \Rightarrow V^{\mathcal{P}}(x, y) \geq 1 - \varepsilon_c. \quad (4)$$

When the above holds with  $\varepsilon_c = 0$ , we say that  $\mathcal{P}$  computes  $f$  with perfect completeness.

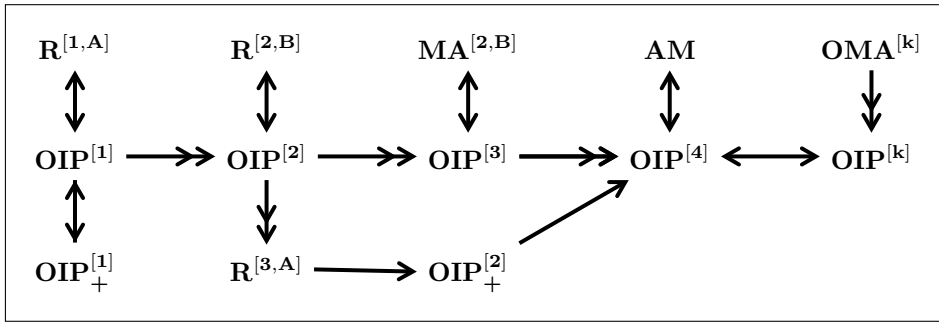
The *verification cost* of  $\mathcal{P}$ , denoted  $\text{vc}(\mathcal{P})$ , is the (worst-case) number of bits sent by Alice plus the number of hidden coin tosses; its *help cost*  $\text{hc}(\mathcal{P})$  is the number of bits communicated between Merlin and Bob; its communication cost  $\text{cc}(\mathcal{P}) = \text{hc}(\mathcal{P}) + \text{vc}(\mathcal{P})$ . For a problem  $f$ , we define its complexity  $\mathbf{C}(f) = \min\{\text{cc}(\mathcal{Q}) : \mathcal{Q} \text{ is a } \mathbf{C} \text{ protocol that solves } f \text{ with } \max\{\varepsilon_s, \varepsilon_c\} \leq 1/3\}$ .

## 4.2 Relations Among Communication Complexity Classes

We prove a number of inclusion and separation results among our “new” communication complexity classes and relate them to previously studied classes. These are summarized in Figure 1, replicated below.

Our results shed light on the landscape of online communication complexity in general.

The simplest online communication model is  $\mathbf{R}^{[1, \mathbf{A}]}$ , a.k.a. one-way randomized communication. The result  $\mathbf{OIP}^{[1]} = \mathbf{OIP}_+^{[1]} = \mathbf{R}^{[1, \mathbf{A}]}$  establishes that in the world of online communication, introducing the omniscient but untrusted Merlin into the model is not enough to obtain super-polynomial efficiency improvements, if *interaction* with Merlin is not permitted. The stronger result that  $\mathbf{OMA}^{[k]} = \mathbf{R}^{[1, \mathbf{A}]}$  for all constants  $k > 0$  (this is the full statement of Theorem 5.20) establishes that in the “public coin” setting, the addition of Merlin is not enough to obtain super-polynomial speedups even if interaction with Merlin is permitted.



■ **Figure 3** The layout of our communication complexity zoo. An arrow from  $C_1$  to  $C_2$  indicates that  $C_1 \subseteq C_2$ . If the arrow is double-headed, then the inclusion is strict. Within the figure,  $k$  is an arbitrary constant larger than 4.

The result that  $OIP^{[2]} = R^{[2,B]}$  (see Corollary 5.7) establishes that in the “hidden coin” setting, the addition of Merlin to the communication model *can* yield super-polynomial efficiency improvements, even if only the barest amount of interaction with Merlin is permitted. However, note that  $R^{[2,B]}$  is the simplest non-online communication model. Thus the combination of hidden coins and a minimal amount of interaction is enough to simulate only the simplest of the non-online communication protocols.

The result that  $OIP^{[4]} = OIP_+^{[4]} = AM$  (see Corollary 5.14) shows that in the “hidden coin” setting, the addition of Merlin to the communication model permits the simulation even of *non-online interactive proofs*, as soon as four rounds of interaction with Merlin are permitted.

This in turn explains the somewhat puzzling result that the  $OIP$  and  $OIP_+$  hierarchies collapse to the fourth level: both Goldwasser–Sipser [16] and Babai–Moran [6] break down in the  $OIP$  and  $OIP_+$  worlds because their transformations *do not preserve online-ness*: they will turn an  $OIP^{[2]}$  protocol into a “public coin” one, but require Merlin to send a message to Alice. However, as soon as four rounds of interaction with Merlin are permitted, even online interactive proofs can simulate non-online ones. At this point, the phenomena of classical interactive proofs kick in, and the hierarchies collapse.

## 5 A Communication Complexity Zoo

We now study our central communication models  $OIP^{[k]}$  and  $OIP_+^{[k]}$ , and prove the web of relationships given in Figure 3. Our results are of two types: (1) establishing separations or collapses between levels of the  $OIP$  and  $OIP_+$  hierarchies, as the case may be, and (2) relating these hierarchies to other previously studied communication complexity classes. We shall first characterize every finite level of the  $OIP$  hierarchy (the vertical bidirectional arrows in Figure 3). Next, in Sections 5.4 and 5.5, we separate the first four levels of the hierarchy (the horizontal double-headed arrows in the figure). Finally, in Section 5.5, we separate the  $OIP$  and  $OMA$  hierarchies.

Throughout Section 5,  $f$  will denote an arbitrary communication problem given by a Boolean function  $f : \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$ , and  $n$  will parametrize its “instance size” up to a constant factor, i.e., we will have  $\log |\mathcal{X}| + \log |\mathcal{Y}| = \Theta(n)$ . We shall use big- $O$  and big- $\Omega$  notation to hide constants independent of  $f$ ,  $|\mathcal{X}|$  and  $|\mathcal{Y}|$ . We shall use the term “ordinary protocol” to mean a randomized communication protocol involving Alice and Bob alone (and no Merlin).

The first level of the hierarchy is easy to characterize.

► **Theorem 5.1.** *We have  $\text{OMA}^{[1]} = \text{OIP}^{[1]} = \text{OIP}_+^{[1]} = \text{MA}^{[1,\mathbf{A}]} = \mathbf{R}^{[1,\mathbf{A}]}$ .*

**Proof.** The definitions immediately show that the first four classes are identical (syntactically) and include  $\mathbf{R}^{[1,\mathbf{A}]}$ , because one can always choose to ignore Merlin. The reverse inclusion  $\text{MA}^{[1,\mathbf{A}]} \subseteq \mathbf{R}^{[1,\mathbf{A}]}$  follows from previous work: Chakrabarti et al. [9] show that for all  $f$  we have  $\mathbf{R}^{[1,\mathbf{A}]}(f) = O(\text{MA}^{[1,\mathbf{A}]}(f)^2)$ . ◀

## 5.1 A Characterization of $\text{OIP}^{[2]}$

The main goal of this subsection is to prove that  $\text{OIP}^{[2]} = \mathbf{R}^{[2,\mathbf{B}]}$ . We start with the following communication analog of Theorem 2.2, which was proven in a streaming setting.

► **Lemma 5.2** (Polynomial Evaluation Protocol, Communication Version). *Suppose Alice holds a  $v$ -variate polynomial  $g$  of total degree  $d$  over a field  $\mathbb{F}$ , and Bob holds a point  $\mathbf{j} \in \mathbb{F}^v$ . Assume  $|\mathbb{F}| > 4d$ . Then there is an  $\text{OIP}^{[2]}$  protocol with communication cost  $O((v+d) \cdot \log |\mathbb{F}|)$  for evaluating  $g(\mathbf{j})$ . In particular,  $\text{OIP}^{[2]}(\text{INDEX}) = O(\log n \log \log n)$ , so that  $\text{INDEX} \in \text{OIP}^{[2]}$ .*

**Proof.** Using the notation established in the description of the polynomial agreement protocol of Section 2.1, we simply note that the hidden coins shared between Alice and Bob determine  $\mathbf{r}$ . Bob can send Merlin (the canonical representation of) the line  $\ell$  that passes through  $\mathbf{j}$  and  $\mathbf{r}$  without having to hear from Alice, since  $\ell$  is determined entirely by  $\mathbf{r}$  and Bob's input  $\mathbf{j}$ . Merlin can send Bob the polynomial  $h(W)$  claimed to equal  $g$  restricted to the line  $\ell$ , and Alice can send Bob  $g(\mathbf{r})$  within the stated cost bounds. Bob performs the same check as the verifier in Theorem 2.2, and the completeness and soundness analysis is unchanged. ◀

The just-proved fact that  $\text{INDEX} \in \text{OIP}^{[2]}$  is striking: combined with the well-known lower bound  $\mathbf{R}^{[1,\mathbf{A}]}(\text{INDEX}) = \Omega(n)$ , it shows that introducing Merlin into the picture while keeping the one-way restriction on the Alice/Bob communication lowers cost exponentially. It is now natural to ask whether  $\text{OIP}^{[2]}$  allows such exponential savings for harder problems, such as DISJ. Our next result – a lower bound on  $\text{OIP}^{[2]}$  complexity – implies that it does not.

► **Theorem 5.3.** *Let  $\mathcal{P}$  be an  $\text{OIP}^{[2]}$  protocol computing  $f$ . Then  $\text{hc}(\mathcal{P}) \text{vc}(\mathcal{P}) = \Omega(\mathbf{R}^{[2,\mathbf{B}]}(f))$ . In particular,  $\text{OIP}^{[2]}(f) = \Omega(\mathbf{R}^{[2,\mathbf{B}]}(f)^{1/2})$ , which implies  $\text{OIP}^{[2]} \subseteq \mathbf{R}^{[2,\mathbf{B}]}$ .*

**Proof.** After appropriate parallel repetition, we may assume that the soundness and completeness errors of  $\mathcal{P}$  at most  $1/12$  each. In general,  $\mathcal{P}$  takes the following shape: (1) hidden coins are tossed, generating random string  $r$  according to distribution  $\mathcal{D}$ ; (2) Bob sends Merlin a message  $m_B = m_B(y, r)$ ; (3) Merlin responds with a message  $m_M = m_M(x, y, m_B)$ ; (4) Alice sends Bob a message  $m_A = m_A(x, r)$ ; (5) Bob outputs a bit given by a function  $\text{out}^{\mathcal{P}}(y, m_M, m_A)$ .

Let  $\mathcal{D}_m$  be  $\mathcal{D}$  conditioned on the event  $\{m_B(y, r) = m\}$ . Note that the distribution  $\mathcal{D}_m$  depends on *both*  $y$  and  $m$ . Since Bob knows  $y$ , Bob can determine the distribution  $\mathcal{D}_m$  for any value of  $m$  (this is not, however, true for Alice, because Alice does not know  $y$ ).

With this notational setup, we now describe (in Figure 4) a two-message ordinary protocol  $\mathcal{Q}$  that we claim computes  $f$ .

To analyze this protocol, let us first define the *weight*  $W_{x,y}(\bar{m})$  of a Bob-message  $\bar{m}$  to be the probability that Merlin, playing optimally after receiving  $\bar{m}$ , convinces Bob to output 1. That is,

$$W_{x,y}(\bar{m}) = \max_{m_M} \Pr_{r \sim \mathcal{D}_{\bar{m}}} \left[ \text{out}^{\mathcal{P}}(y, m_M, m_A(x, r)) = 1 \right]. \quad (5)$$

1. Bob samples  $\bar{r} \sim \mathcal{D}$ , computes  $\bar{m} = m_B(y, \bar{r})$ , then sends Alice i.i.d. samples  $r^{(1)}, \dots, r^{(h)} \sim \mathcal{D}_{\bar{m}}$ , where  $h = 36(\text{hc}(\mathcal{P}) + 4)$ .
2. Alice sends Bob  $m_A(x, r^{(1)}), \dots, m_A(x, r^{(h)})$ .
3. Bob outputs 1 iff  $\exists m_M : |\{i \in [h] : \text{out}^{\mathcal{P}}(y, m_M, m_A(x, r^{(i)})) = 1\}| > h/2$ .

■ **Figure 4** The  $\mathbf{R}^{[2, \mathbf{B}]}$  protocol  $\mathcal{Q}$ , which simulates the  $\mathbf{OIP}^{[2]}$  protocol  $\mathcal{P}$ .

Then, with  $\bar{m} \sim m_B(y, \mathcal{D})$ , the expected weight  $\mathbb{E}_{\bar{m}}[W_{x,y}(\bar{m})]$  is at least  $11/12$  when  $f(x, y) = 1$  and at most  $1/12$  when  $f(x, y) = 0$ .

**Correctness on 1-inputs:** Fix  $(x, y) \in f^{-1}(1)$ . We shall proceed assuming that the specific Bob-message  $\bar{m}$  chosen in Step 1 of  $\mathcal{Q}$  satisfies  $W_{x,y}(\bar{m}) > 2/3 = 1 - 4(1/12)$ ; by Markov’s inequality, this fails to happen with probability at most  $1/4$ . Studying Eq. (5) tell us that there exists a specific Merlin-message  $m_M^*$  such that  $\Pr_r[\text{out}^{\mathcal{P}}(y, m_M^*, m_A(x, r)) = 1] > 2/3$ . Therefore, according to the strategy in Steps 2 and 3, the size of the set  $\{i \in [h] : \text{out}^{\mathcal{P}}(y, m_M^*, m_A(x, r^{(i)})) = 1\}$  is a sum of  $h$  i.i.d. indicators and exceeds  $2h/3$  in expectation. By standard Chernoff bounds (e.g., [28, Theorem 4.4]), the probability that Bob outputs 0 is  $2^{-\Omega(h)}$ . Thus, overall, the probability that  $\mathcal{Q}$  outputs 0 on input  $(x, y)$  is at most  $1/4 + 2^{-\Omega(h)} < 1/3$ .

**Correctness on 0-inputs:** Fix  $(x, y) \in f^{-1}(0)$ . We shall proceed assuming that the specific Bob-message  $\bar{m}$  chosen in Step 1 of  $\mathcal{Q}$  satisfies  $W_{x,y}(\bar{m}) < 1/3$ ; by Markov’s inequality, this fails to happen with probability at most  $1/4$ . For each specific Merlin-message  $m_M$ , define

$$\text{size}(m_M) = \left| \{i \in [h] : \text{out}^{\mathcal{P}}(y, m_M, m_A(x, r^{(i)})) = 1\} \right|.$$

Then  $\text{size}(m_M)$  is a sum of  $h$  i.i.d. indicators and has expectation below  $h/3$ . By standard Chernoff bounds,  $\Pr[\text{size}(m_M) > h/2] \leq e^{-h/36}$ . By a union bound over all possible Merlin-messages  $m_M$ , the probability that Bob outputs 1 is at most  $2^{\text{hc}(\mathcal{P})} e^{-h/36} < 2^{-4}$ , using our choice of  $h$ . Adding in the  $1/4$  from our Markov argument earlier, the overall probability that  $\mathcal{Q}$  outputs 1 on input  $(x, y)$  is at most  $1/4 + 2^{-4} < 1/3$ .

**Communication Cost:** By definition of the  $\mathbf{OIP}^{[2]}$  model, we have  $|r| \leq \text{vc}(\mathcal{P})$  and  $|m_A| \leq \text{vc}(\mathcal{P})$ . Thus, each of the two messages in  $\mathcal{Q}$  costs at most  $h \cdot \text{vc}(\mathcal{P}) = O(\text{hc}(\mathcal{P}) \text{vc}(\mathcal{P}))$  bits. ◀

The above proof exploits a key property of  $\mathbf{OIP}^{[2]}$  protocols: Bob can sample from the conditional distribution  $\mathcal{D}_{\bar{m}}$ . This is possible because  $m_B = m_B(y, r)$  is independent of Alice’s message  $m_A$ , a property not satisfied in the stronger  $\mathbf{OIP}_+^{[2]}$  model. This explains why Theorem 5.3 does not apply to  $\mathbf{OIP}_+^{[2]}$ , and indeed we shall later give an exponential separation between  $\mathbf{OIP}^{[2]}$  and  $\mathbf{OIP}_+^{[2]}$  in Corollary 5.19.

Theorem 5.3 implies a number of lower bounds for specific problems. We begin with DISJ.

► **Corollary 5.4.** *We have  $\Omega(n^{1/2}) \leq \mathbf{OIP}^{[2]}(\text{DISJ}) \leq O(n^{1/2} \log n)$ . In particular,  $\text{DISJ} \notin \mathbf{OIP}^{[2]}$ .*

**Proof.** For the lower bound, we combine Theorem 5.3 with the fact that  $\mathbf{R}^{[2, \mathbf{B}]}(\text{DISJ}) \geq \mathbf{R}(\text{DISJ}) = \Omega(n)$ , the last step being a celebrated lower bound [21]. The upper bound follows from the Aaronson–Wigderson protocol [2] for DISJ, which is in fact an  $\mathbf{MA}^{[1, \mathbf{A}]}$  protocol. ◀

We remark that we may replace DISJ in Corollary 5.4 with IP2, the “inner product mod 2” function. Indeed, the Aaronson–Wigderson protocol also applies to IP2, and  $R(\text{IP2}) = \Omega(n)$ .

Recall that MED is a relation on inputs in  $[n]^m \times [n]^m$ . Corollary 5.5 below establishes a lower bound of  $\Omega(m^{1/4})$  on the cost of any  $\mathbf{OIP}^{[2]}$  protocol for MED (the proof is deferred to the full version of the paper). This justifies our use of three rounds in the polylogarithmic cost SIP for MEDIAN we gave in Theorem 3.7, as it implies that any 2-round SIP for median based on known techniques must have polynomial cost.

► **Corollary 5.5.** *We have  $\Omega(m^{1/4}) \leq \mathbf{OIP}^{[2]}(\text{MED}) \leq O(m^{1/2} \log^{3/2} n)$ .*

We have now seen that up to polynomial (specifically, quadratic) blowup,  $\mathbf{OIP}^{[2]}$  is no more powerful than ordinary  $\mathbf{R}^{[2,B]}$ . We now show that up to another quadratic blowup this is in fact a characterization.

► **Theorem 5.6.** *For all  $f$ , we have  $\mathbf{OIP}^{[2]}(f) = O(\mathbf{R}^{[2,B]}(f)^2)$ . In particular,  $\mathbf{OIP}^{[2]} \supseteq \mathbf{R}^{[2,B]}$ .*

**Proof.** Let  $\mathcal{Q}$  be an  $\mathbf{R}^{[2,B]}$  protocol for  $f$  with cost  $C$  and error at most  $1/6$ . Assume WLOG that  $C \geq 5$  and that each of the two messages in  $\mathcal{Q}$  is a string in  $\{0, 1\}^C$ . We shall treat Alice’s messages as elements of the field  $\mathbb{F} = \mathbb{F}_{2^C}$  via an agreed-upon bijection.

We design an  $\mathbf{OIP}^{[2]}$  protocol  $\mathcal{P}$  for  $f$ , based on  $\mathcal{Q}$ . Given an input  $(x, y)$ ,  $\mathcal{P}$  begins by choosing a (hidden) random string  $r$  shared between Alice and Bob exactly as  $\mathcal{Q}$  would have. From now on, think of  $x, y, r$  as fixed. This then fixes a message  $m_B$  that Bob would have sent Alice in  $\mathcal{Q}$ , as well as a function  $m_A : \{0, 1\}^C \rightarrow \mathbb{F}$  specifying Alice’s response to each Bob-message. Let  $\tilde{m}_A(Z_1, \dots, Z_C) \in \mathbb{F}[Z_1, \dots, Z_C]$  be the multilinear extension of this function  $m_A$ . In  $\mathcal{P}$ , Alice needs to send a message to Bob that allows him to determine  $m_A(m_B) = \tilde{m}_A(m_B)$  with Merlin’s help. This is an instance of polynomial evaluation, so we solve it by applying the  $\mathbf{OIP}^{[2]}$  polynomial evaluation protocol (PEP) from Lemma 5.2.

The polynomial  $\tilde{m}_A$  is  $C$ -variate and has total degree  $C$ . Therefore, PEP has communication cost  $O(C \log |\mathbb{F}|) = O(C^2)$ , as does  $\mathcal{P}$ . Next, PEP has perfect completeness, so an honest Merlin can cause  $\mathcal{P}$  to output 1 whenever the choice of  $r$  would have caused  $\mathcal{Q}$  to output 1. Finally, PEP has soundness error at most  $C/(|\mathbb{F}| - 1) = C/(2^C - 1) < 1/6$ , so a dishonest Merlin can cause  $\mathcal{P}$  to differ in output from  $\mathcal{Q}$  with probability at most  $1/6$ . Using the error bound of  $1/6$  on  $\mathcal{Q}$ , we conclude that  $\mathcal{P}$  has completeness error at most  $1/6$  and soundness error at most  $1/6 + 1/6 = 1/3$ . ◀

► **Corollary 5.7.** *For all  $f$ , we have  $\Omega(\mathbf{R}^{[2,B]}(f)^{1/2}) \leq \mathbf{OIP}^{[2]}(f) \leq O(\mathbf{R}^{[2,B]}(f)^2)$ . Thus,  $\mathbf{OIP}^{[2]} = \mathbf{R}^{[2,B]}$ .*

**Proof.** Combine Theorems 5.3 and 5.6. ◀

## 5.2 A Characterization of $\mathbf{OIP}^{[3]}$

The main goal of this subsection is to prove that  $\mathbf{OIP}^{[3]} = \mathbf{MA}^{[2,B]}$ . Theorem 5.8 below gives a lower bound that builds on the argument in Theorem 5.3 (the proof is deferred to the full version of the paper). Just as before, we can then derive a lower bound for the specific problem DISJ.

► **Theorem 5.8.** *Let  $\mathcal{P}$  be an  $\mathbf{OIP}^{[3]}$  protocol computing  $f$ . Then there is an  $\mathbf{MA}^{[2,B]}$  protocol  $\mathcal{Q}$  computing  $f$  with  $\text{hc}(\mathcal{Q}) \leq \text{hc}(\mathcal{P})$  and  $\text{vc}(\mathcal{Q}) = O(\text{hc}(\mathcal{P}) \text{vc}(\mathcal{P}))$ . In particular,  $\mathbf{OIP}^{[3]}(f) = \Omega(\mathbf{MA}^{[2,B]}(f)^{1/2})$ , which implies  $\mathbf{OIP}^{[3]} \subseteq \mathbf{MA}^{[2,B]}$ .*

► **Corollary 5.9.** *We have  $\Omega(n^{1/3}) \leq \text{OIP}^{[3]}(\text{DISJ}) \leq O(n^{1/3} \log n)$ . In particular,  $\text{DISJ} \notin \text{OIP}^{[3]}$ .*

**Proof.** Klauck [22] proved that  $\text{MA}(\text{DISJ}) = \Omega(n^{1/2})$ . Applying Theorem 5.8 to this result gives the non-tight bound  $\text{OIP}^{[3]}(\text{DISJ}) = \Omega(n^{1/4})$ . But we observe that Klauck’s proof shows something stronger: namely, if an **MA** protocol  $\mathcal{Q}$  computes  $\text{DISJ}$ , then  $\text{hc}(\mathcal{Q}) \text{vc}(\mathcal{Q}) = \Omega(n)$ . Combining Theorem 5.8 with *this* result, we conclude that if an **OIP**<sup>[3]</sup> protocol  $\mathcal{P}$  computes  $\text{DISJ}$ , then  $\text{hc}(\mathcal{P})^2 \text{vc}(\mathcal{P}) = \Omega(n)$ , and therefore  $\text{hc}(\mathcal{P}) + \text{vc}(\mathcal{P}) = \Omega(n^{1/3})$ .

For the upper bound, we note that Aaronson and Wigderson [2] also gave an online **MAMA** protocol for  $\text{DISJ}$  of cost  $O(n^{1/3} \log n)$ . Every online **MAMA** protocol admits a simulation in **OIP**<sup>[3]</sup>. ◀

As with Corollary 5.4, we may replace  $\text{DISJ}$  in the above result with  $\text{IP2}$ . Indeed, Klauck’s result [22] implies that  $\text{MA}(\text{IP2}) = \Omega(n^{1/2})$ , and Aaronson and Wigderson’s **MAMA** protocol also applies to  $\text{IP2}$ .

As we did for the second level in the **OIP** hierarchy, we give an upper bound that applies to the third level and gives a characterization that is tight up to a quadratic blowup.

► **Theorem 5.10.** *For all  $f$ , we have  $\text{OIP}^{[3]}(f) = O(\text{MA}^{[2,B]}(f)^2)$ . In particular,  $\text{OIP}^{[3]} \supseteq \text{MA}^{[2,B]}$ .*

**Proof sketch.** We build on the argument in Theorem 5.6 exactly as the proof of Theorem 5.8 builds on Theorem 5.3. Given an **MA**<sup>[2,B]</sup> protocol  $\mathcal{Q}$  of cost  $C$ , the verification strategy used by Alice and Bob in  $\mathcal{Q}$  is an **R**<sup>[2,B]</sup> protocol of cost  $C$ , which we can replace with an **OIP**<sup>[2]</sup> protocol of cost  $O(C^2)$ , by Theorem 5.6. After this replacement we have an **OIP**<sup>[3]</sup> protocol. The remaining analysis is routine. ◀

► **Corollary 5.11.** *For all  $f$ ,  $\Omega(\text{MA}^{[2,B]}(f)^{1/2}) \leq \text{OIP}^{[3]}(f) \leq O(\text{MA}^{[2,B]}(f)^2)$ . Thus,  $\text{OIP}^{[3]} = \text{MA}^{[2,B]}$ .*

**Proof.** Combine Theorems 5.8 and 5.10. ◀

### 5.3 A Characterization of **OIP**<sup>[4]</sup> and Beyond

The fourth level of the **OIP** hierarchy turns out to have surprising power. It can capture all of **AM**, a model that lies at the frontier of our current understanding of communication complexity classes in the sense that we do not know any nontrivial **AM** lower bounds. Thanks to this surprising power, we can show that all constant-height levels of the **OIP** hierarchy collapse to the fourth level.

► **Theorem 5.12.** *For all  $f$ , we have  $\text{OIP}^{[4]}(f) = O(\text{AM}(f) \log \text{AM}(f))$ . In particular,  $\text{OIP}^{[4]} \supseteq \text{AM}$ .*

**Proof.** Suppose  $\text{AM}(f) = C$ . WLOG, there is a protocol  $\mathcal{Q}$  for  $f$  with the following shape: Bob tosses coins to generate a random string  $r$  and sends it to Merlin, who responds with a message  $m$ , where  $|r| + |m| \leq C$ . Bob then sends  $(r, m)$  to Alice, who responds with a single bit, after which Bob announces the output.

The interaction between Bob and Alice is an **R**<sup>[2,B]</sup> protocol (in fact, it is deterministic) of cost  $C$ . Theorem 5.6 shows that it can be replaced with an **OIP**<sup>[2]</sup> protocol of cost  $O(C^2)$ . Performing this replacement gives us an **OIP**<sup>[4]</sup> protocol for  $f$ . The cost bound can be improved to  $O(C \log C)$  by revisiting the analysis of the polynomial evaluation protocol used to prove Theorem 5.6 and using the fact that Alice’s message in  $\mathcal{Q}$  is just a single bit. ◀

► **Theorem 5.13.** *For each  $k > 0$ , there exists a constant  $c_k > 0$  such that for all  $f$ ,  $\text{OIP}_+^{[k]}(f) \geq \Omega(\text{AM}(f)^{c_k})$ . In particular, for every constant  $k$ , we have  $\text{OIP}_+^{[k]} \subseteq \text{AM}$ .*

**Proof.** Let  $C = \text{OIP}_+^{[k]}(f)$  and let  $\mathcal{P}$  be an  $\text{OIP}_+^{[k]}$  protocol with cost  $C$  that computes  $f$ . By definition,  $\mathcal{P}$  uses a hidden random string and Merlin learns about this string only indirectly, from Bob’s computed messages. We apply the Goldwasser–Sipser set lower bound technique [16] to convert  $\mathcal{P}$  into a protocol where all random coins are directly revealed. Specifically, we can convert  $\mathcal{P}$  into an  $\text{AMAM} \cdots \text{AM}$  protocol  $\mathcal{Q}'$ , where  $k + 3$  messages are sent in total: Merlin’s messages are broadcast and after his final message Alice sends a message to Bob, who announces the output. We have  $\text{cc}(\mathcal{Q}') = O(C^{a_k})$  for some constant  $a_k \geq 1$ .

We apply Babai and Moran’s round elimination techniques [6] to turn  $\mathcal{Q}'$  into a standard  $\text{AM}$  protocol  $\mathcal{Q}$  of cost at most  $O(\text{cc}(\mathcal{Q}')^{b_k})$  for some constant  $b_k \geq 1$ . The result follows by taking  $c_k = 1/(a_k b_k)$ . ◀

► **Corollary 5.14.** *For all  $f$ ,  $\Omega(\text{AM}(f)^{c_4}) \leq \text{OIP}^{[4]}(f) \leq O(\text{AM}(f) \log \text{AM}(f))$ , where  $c_4$  is the constant from Theorem 5.13. In particular,  $\text{OIP}^{[4]} = \text{AM}$ , and in fact  $\text{OIP}^{[k]} = \text{AM}$  for every constant  $k \geq 4$ .*

**Proof.** Combine Theorems 5.13 and 5.12, noting that  $\text{OIP}^{[k]} \subseteq \text{OIP}_+^{[k]}$  for every  $k \geq 4$ . ◀

Here is an interesting point worth contemplating. On the one hand, our transformations in the proof of Theorem 5.13 perform round reduction at the expense of destroying online-ness: the final protocol  $\mathcal{Q}$  is no longer online, i.e., we cannot require communications to go to Bob alone. On the other hand, the transformation in the proof of Theorem 5.12 “restores” onlineness at only a “slight” expense of requiring four rounds, whereas  $\text{AM}$  uses only two. Overall, we have a collapse of the  $\text{OIP}$  hierarchy to its fourth level.

We have also noted earlier that we (regretfully) do not yet know how to place a concrete problem outside  $\text{OIP}_+^{[2]}$ . Nevertheless, Theorems 5.12 and 5.13 together establish a weakness of  $\text{OIP}_+^{[2]}$ : up to polynomial factors this model is no more powerful than  $\text{OIP}^{[4]}$ .

## 5.4 Exponential Separations in Our Complexity Zoo

Among the first four levels of the  $\text{OIP}$  hierarchy, we can now show that every pair of adjacent levels is exponentially separated. The next three results make this precise. Recall that  $\text{INTER} = \neg\text{DISJ}$  is the set intersection problem.

► **Theorem 5.15.** *We have  $\text{OIP}^{[1]}(\text{INDEX}) = \Omega(n^{1/2})$  whereas  $\text{OIP}^{[2]}(\text{INDEX}) = O(\log n \log \log n)$ .*

**Proof.** Combine Theorems 5.1 and Lemma 5.2, and then the known results that  $\text{MA}^{[1,A]}(f) = \Omega(\text{R}^{[1,A]}(f)^{1/2})$  for all  $f$  [9] (see also Theorem 5.20 in Section 5.5), and that  $\text{R}^{[1,A]}(\text{INDEX}) = \Omega(n)$  [3]. ◀

► **Theorem 5.16.** *We have  $\text{OIP}^{[2]}(\text{INTER}) = \Omega(n^{1/2})$  whereas  $\text{OIP}^{[3]}(\text{INTER}) = O(\log^2 n)$ .*

**Proof.** For the lower bound, use  $\text{R}^{[2,B]}(\text{INTER}) \geq \text{R}(\text{INTER}) = \text{R}(\text{DISJ}) = \Omega(n)$  and then apply Theorem 5.3.

For the upper bound, note that  $\text{INTER}$  has a nondeterministic protocol with cost  $O(\log n)$ , wherein Alice and Bob guess an element in the intersection of their respective sets and they verify membership. In particular this gives  $\text{MA}^{[2,B]}(\text{INTER}) = O(\log n)$ ; in fact, Bob need not send anything to Alice in the  $\text{MA}^{[2,B]}$  protocol. Now apply Theorem 5.10. ◀



While we do not know of a *total* Boolean function that separates  $\mathbf{OIP}^{[3]}$  from  $\mathbf{OIP}^{[4]}$ , we do know of a *partial* Boolean function whose  $\mathbf{OIP}^{[3]}$  communication complexity is exponentially larger than its  $\mathbf{OIP}^{[4]}$  communication complexity. Specifically, Klauck [23, Corollary 3] gives a promise problem he calls PAPPMP which has *Quantum* Merlin-Arthur ( $\mathbf{QMA}$ ) communication complexity  $\Omega(n^{1/6})$  and  $\mathbf{AM}$  communication complexity  $O(\log n)$ . Since Theorem 5.8 shows that any  $\mathbf{OIP}^{[3]}$  protocol can be transformed into an equivalent  $\mathbf{MA}^{[2,B]}$  protocol with a quadratic blowup in cost, and  $\mathbf{MA}^{[2,B]}$  protocols are simply restricted versions of  $\mathbf{QMA}$  protocols, Klauck's lower bound on the  $\mathbf{QMA}$  cost of PAPPMP implies that  $\mathbf{OIP}^{[3]}(\text{PAPPMP}) = \Omega(n^{1/12})$ .

Meanwhile, Theorem 5.12 shows that any  $\mathbf{AM}$  communication protocol can be transformed into an equivalent  $\mathbf{OIP}^{[4]}$  protocol with a logarithmic blowup in costs. Thus, Klauck's upper bound on the  $\mathbf{AM}$  communication complexity of PAPPMP implies that  $\mathbf{OIP}^{[4]}(\text{PAPPMP}) = O(\log n \log \log n)$ .

► **Theorem 5.17.** *We have  $\mathbf{OIP}^{[3]}(\text{PAPPMP}) = \Omega(n^{1/12})$  whereas  $\mathbf{OIP}^{[4]}(\text{PAPPMP}) = O(\log n \log \log n)$ .*

Next, we show that, up to polynomial factors,  $\mathbf{OIP}_+^{[2]}$  is at least as powerful as  $\mathbf{R}^{[3,A]}$ , the class of *three-message* randomized communication protocols in which Alice speaks first. This will enable us to exhibit an explicit function  $f$  on domain  $\{-1, 1\}^n \times \{-1, 1\}^n$  such that  $\mathbf{OIP}^{[2]}(f) = \Omega(\sqrt{n/\log n})$ , while  $\mathbf{OIP}_+^{[2]}(f) = O(\log^2 n)$ .

► **Theorem 5.18.** *For all  $f$ , we have  $\mathbf{OIP}_+^{[2]}(f) = O(\mathbf{R}^{[3,A]}(f)^2)$ .*

**Proof.** Let  $\mathcal{Q}$  be any three-message randomized communication protocol of cost  $C$ , with Alice speaking first. We show how to convert  $\mathcal{Q}$  into an  $\mathbf{OIP}_+^{[2]}$  protocol  $\mathcal{P}$  of cost  $O(C^2)$ .

We think of  $\mathcal{Q}$  as consisting of one message  $m_A^{(1)}$  from Alice to Bob, followed by a *two-message* communication protocol  $\mathcal{Q}'$  in which Bob speaks first. Theorem 5.6 shows how to transform  $\mathcal{Q}'$  into an equivalent  $\mathbf{OIP}^{[2]}$  protocol  $\mathcal{P}'$  of cost  $O(C^2)$  (note this  $\mathbf{OIP}^{[2]}$  protocol depends on  $m_A^{(1)}$ ).

Thus, we obtain an  $\mathbf{OIP}_+^{[2]}$  protocol  $\mathcal{P}$  as follows. Alice's message to Bob in  $\mathcal{P}$  consists of two parts. The first specifies  $m_A^{(1)}$ , and the second is the message she would have sent to Bob in  $\mathcal{P}'$ . Bob, who learns  $m_A^{(1)}$  from the first part of Alice's message, now knows what  $\mathbf{OIP}^{[2]}$  protocol  $\mathcal{P}'$  to execute, and simply behaves the same as he would in  $\mathcal{P}'$ . ◀

Exponential separations between  $\mathbf{R}^{[3,A]}$  and  $\mathbf{R}^{[2,B]}$  are known. In particular, consider the  $k$ -step (bipartite) pointer jumping function  $\text{PJ}_k$ , which interprets each of Alice and Bob's inputs as a list of  $N = \Theta(n/\log n)$  *pointers*, a pointer being a  $(\log N)$ -bit integer. Each pointer in a player's list is interpreted as pointing to (i.e., indexing) a pointer in the other player's list. The goal is to follow these pointers, starting at the first pointer in Alice's list, and output the  $k$ th pointer encountered. For example, if Alice's input is  $x = (00, 01, 10, 00)$  and Bob's input is  $y = (01, 10, 11, 00)$ , then  $\text{PJ}_1(x, y) = 01$ ,  $\text{PJ}_2(x, y) = 01$ ,  $\text{PJ}_3(x, y) = 10$ , and so on. To turn  $\text{PJ}_k$  into a Boolean function  $\text{BPJ}_k$ , we take the parity of the  $(\log N)$ -bit output of  $\text{PJ}_k$ .

► **Corollary 5.19.** *We have  $\mathbf{OIP}^{[2]}(\text{BPJ}_2) = \Omega(\sqrt{n/\log n})$ , while  $\mathbf{OIP}_+^{[2]}(\text{BPJ}_2) = O(\log^2 n)$ .*

**Proof.** Nisan and Wigderson [30] showed that  $\mathbf{R}^{[k,B]}(\text{BPJ}_k) = \Omega(N/k^2 - k \log N)$ . In particular, any two-message randomized communication protocol in which Bob speaks first has cost  $\Omega(N)$ . Hence, Theorem 5.3 implies that  $\mathbf{OIP}^{[2]}(\text{BPJ}_2) = \Omega(\sqrt{n/\log n})$ .

To prove the upper bound on  $\text{OIP}_+^{[2]}(\text{BPJ}_2)$ , note that there is a trivial three-message protocol for  $\text{PJ}_2$  (and hence for  $\text{BPJ}_2$ ) of cost  $O(\log n)$  in which Alice speaks first. Now apply Theorem 5.18. ◀

## 5.5 An Exponential Separation Between $\text{OIP}^{[2]}$ and $\text{OMA}^{[k]}$

Theorem 5.20 establishes that for any function  $f$ ,  $\text{OMA}^{[2k]}(f) = \Omega(\mathbb{R}^{[1,A]}(f)^{1/(k+1)})$ . An essentially identical lower bound was proven by Klauck and Prakash for a closely related (though not identical) communication model; the full version of the paper provides a detailed proof for completeness, and in the process identifies the crucial details of the communication model that enable the lower bound to hold.

► **Theorem 5.20.** *For any function  $f$  and constant  $k$ ,  $\text{OMA}^{[2k]}(f) = \Omega(\mathbb{R}^{[1,A]}(f)^{1/(k+1)})$ .*

The main property of the  $\text{OMA}^{[k]}$  communication model exploited in our proof of Theorem 5.20 is the following: in any  $\text{OMA}^{[k]}$  protocol  $\mathcal{P}$ , for all  $i \leq k$ , Alice can determine Bob’s  $i$ th message to Merlin in  $\mathcal{P}$  on her own. In particular, the same lower bound would apply to any variant of online Arthur-Merlin communication models in which Bob’s messages to Merlin must be independent of his input  $y$ . This is the intuitive reason why the  $\text{OIP}^{[2]}$  model is exponentially more powerful than the  $\text{OMA}^{[k]}$  model for any constant  $k$ : in the  $\text{OIP}^{[2]}$  model, Bob’s message to Merlin may depend on his input  $y$ , while this is not allowed in the  $\text{OMA}^{[k]}$  model.

Combining Theorem 5.20 with Lemma 5.2, which says that  $\text{OIP}^{[2]}(\text{INDEX}) = O(\log n \log \log n)$ , we obtain an exponential separation between  $\text{OIP}^{[2]}$  and  $\text{OMA}^{[k]}$  for any constant  $k > 0$ .

► **Corollary 5.21.** *For every constant  $k > 0$ , we have  $\text{OIP}^{[2]} \not\subseteq \text{OMA}^{[k]}$ .* ◀

## 6 Conclusion

Our primary objects of study in this paper were constant-round interactive protocols for verifying outsourced streaming computations. Our main algorithmic contributions were to give constant-round streaming interactive proofs for a large class of “query” problems. Our protocols are exponentially more efficient than what was believed possible based on prior work, and demonstrate that in the streaming setting, “hidden” coins are exponentially more powerful than public coins.

We also introduced new “online” communication hierarchies,  $\text{OIP}_+$  and  $\text{OIP}$ , which can be seen as restricted variants of the standard Arthur-Merlin communication model. The flow of information in the  $\text{OIP}_+$  and  $\text{OIP}$  models is severely restricted (neither Bob nor Merlin can speak to Alice), yet  $\text{OIP}_+$  is still powerful enough to simulate any streaming interactive proof, and  $\text{OIP}$  powerful enough to simulate all *known* streaming interactive proofs. Our study revealed that the online nature of these communication models leads them to behave very differently from classical interactive proofs, and allowed us to establish strong limitations on the power of existing techniques for developing constant-round SIPs. It also yielded a surprising characterization of the communication complexity class  $\mathbf{AM}$  in terms of online communication models (namely,  $\mathbf{AM} = \text{OIP}^{[4]} = \text{OIP}_+^{[4]}$ ). We believe this characterization may prove useful in establishing non-trivial  $\mathbf{AM}$  lower bounds, a problem that has been identified [23] as an important “first step” toward resolving the  $\mathbf{\Pi}_2 \neq \mathbf{\Sigma}_2$  problem in two-party communication complexity, one of the most important problems left open by Babai et al. [5].

Many questions remain for future work, but here we highlight just one: proving a superlogarithmic lower bound on the  $\mathbf{OIP}_+^{[2]}$  communication cost of an explicit function. Progress on this question would yield the first superlogarithmic lower bounds on the cost of two-round SIPs. Moreover, we have shown that standard techniques easily establish that  $\mathbf{OIP}_+^{[2]}$  is a subset of  $\mathbf{AM}$ , but have been unable to prove any superlogarithmic lower bounds against  $\mathbf{OIP}_+^{[2]}$  protocols. Proving  $\mathbf{OIP}_+^{[2]}$  lower bounds therefore represents an important (and potentially tractable) “zeroth step” toward resolving  $\mathbf{\Pi}_2 \neq \mathbf{\Sigma}_2$ .

**Acknowledgments.** This work was performed while the authors were visiting the Simons Institute for the Theory of Computing. Amit Chakrabarti was supported in part by NSF grant CCF-1217375. Graham Cormode was supported in part by the Yahoo Research Faculty Engagement Program, a Royal Society Wolfson Research Merit Award, and European Research Council grant ERC-CoG-647557. Andrew McGregor and Suresh Venkatasubramanian were supported in part by NSF grant IIS-1251110. Justin Thaler was supported by a Research Fellowship from the Simons Institute for the Theory of Computing.

---

## References

- 1 Scott Aaronson.  $\text{Qma}/\text{qpoly} \subseteq \text{pspace}/\text{poly}$ : De-merlinizing quantum protocols. In *21st Annual IEEE Conference on Computational Complexity (CCC 2006), 16-20 July 2006, Prague, Czech Republic*, pages 261–273. IEEE Computer Society, 2006.
- 2 Scott Aaronson and Avi Wigderson. Algebrization: A new barrier in complexity theory. *TOCT*, 1(1), 2009.
- 3 Farid Ablayev. Lower bounds for one-way probabilistic communication complexity and their application to space complexity. *Theoretical Computer Science*, 175(2):139–159, 1996.
- 4 Noga Alon, Yossi Matias, and Mario Szegedy. The space complexity of approximating the frequency moments. *J. Comput. Syst. Sci.*, 58(1):137–147, 1999.
- 5 Laszlo Babai, Peter Frankl, and Janos Simon. Complexity classes in communication complexity theory. In *Foundations of Computer Science, 1986., 27th Annual Symposium on*, pages 337–347, Oct 1986.
- 6 László Babai and Shlomo Moran. Arthur-merlin games: A randomized proof system, and a hierarchy of complexity class. *J. Comput. Syst. Sci.*, 36(2):254–276, April 1988.
- 7 Dan Boneh, Giovanni Di Crescenzo, Rafail Ostrovsky, and Giuseppe Persiano. Public key encryption with keyword search. In *Advances in Cryptology – EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings*, volume 3027 of *Lecture Notes in Computer Science*, pages 506–522. Springer, 2004.
- 8 Amit Chakrabarti, Graham Cormode, Navin Goyal, and Justin Thaler. Annotations for sparse data streams. In Chandra Chekuri, editor, *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 687–706. SIAM, 2014.
- 9 Amit Chakrabarti, Graham Cormode, Andrew McGregor, and Justin Thaler. Annotations in data streams. *ACM Transactions on Algorithms*, 11(1):7, 2014. A preliminary version of this paper by A. Chakrabarti, G. Cormode, and A. McGregor appeared in *ICALP 2009*.
- 10 Kai-Min Chung, Yael Tauman Kalai, Feng-Hao Liu, and Ran Raz. Memory delegation. In Phillip Rogaway, editor, *Advances in Cryptology – CRYPTO 2011 – 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings*, volume 6841 of *Lecture Notes in Computer Science*, pages 151–168. Springer, 2011.
- 11 Raphaël Clifford, Klim Efremenko, Ely Porat, and Amir Rothschild. From coding theory to efficient pattern matching. In Claire Mathieu, editor, *Proceedings of the Twentieth*

- Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2009, New York, NY, USA, January 4-6, 2009*, pages 778–784. SIAM, 2009.
- 12 A. Condon. The complexity of space bounded interactive proof systems, 1993.
  - 13 Graham Cormode, Michael Mitzenmacher, and Justin Thaler. Practical verified computation with streaming interactive proofs. In Shafi Goldwasser, editor, *Innovations in Theoretical Computer Science 2012, Cambridge, MA, USA, January 8-10, 2012*, pages 90–112. ACM, 2012.
  - 14 Graham Cormode, Michael Mitzenmacher, and Justin Thaler. Streaming graph computations with a helpful advisor. *Algorithmica*, 65(2):409–442, 2013.
  - 15 Graham Cormode, Justin Thaler, and Ke Yi. Verifying computations with streaming interactive proofs. *PVLDB*, 5(1):25–36, 2011.
  - 16 S Goldwasser and M Sipser. Private coins versus public coins in interactive proof systems. In *Proceedings of the Eighteenth Annual ACM Symposium on Theory of Computing, STOC’86*, pages 59–68, New York, NY, USA, 1986. ACM.
  - 17 Shafi Goldwasser, Dan Gutfreund, Alexander Healy, Tali Kaufman, and Guy N. Rothblum. A (de)constructive approach to program checking. In *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing, STOC’08*, pages 143–152, New York, NY, USA, 2008. ACM.
  - 18 Shafi Goldwasser, Yael Tauman Kalai, and Guy N. Rothblum. Delegating computation: Interactive proofs for muggles. In *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing, STOC’08*, pages 113–122, New York, NY, USA, 2008. ACM.
  - 19 Tom Gur and Ran Raz. Arthur-merlin streaming complexity. In Fedor V. Fomin, Rusins Freivalds, Marta Kwiatkowska, and David Peleg, editors, *Automata, Languages, and Programming*, volume 7965 of *Lecture Notes in Computer Science*, pages 528–539. Springer Berlin Heidelberg, 2013.
  - 20 Adam Kalai. Efficient pattern-matching with don’t cares. In David Eppstein, editor, *Proceedings of the Thirteenth Annual ACM-SIAM Symposium on Discrete Algorithms, January 6-8, 2002, San Francisco, CA, USA.*, pages 655–656. ACM/SIAM, 2002.
  - 21 Bala Kalyanasundaram and Georg Schintger. The probabilistic communication complexity of set intersection. *SIAM J. Discret. Math.*, 5(4):545–557, November 1992.
  - 22 Hartmut Klauck. Rectangle size bounds and threshold covers in communication complexity. In *18th Annual IEEE Conference on Computational Complexity (Complexity 2003), 7-10 July 2003, Aarhus, Denmark*, pages 118–134. IEEE Computer Society, 2003.
  - 23 Hartmut Klauck. On arthur merlin games in communication complexity. In *Proceedings of the 26th Annual IEEE Conference on Computational Complexity, CCC 2011, San Jose, California, June 8-10, 2011*, pages 189–199. IEEE Computer Society, 2011.
  - 24 Hartmut Klauck and Ved Prakash. Streaming computations with a loquacious prover. In *Proceedings of the 4th Conference on Innovations in Theoretical Computer Science, ITCS’13*, pages 305–320, New York, NY, USA, 2013. ACM.
  - 25 Ilan Kremer, Noam Nisan, and Dana Ron. On randomized one-round communication complexity. In *Proceedings of the Twenty-seventh Annual ACM Symposium on Theory of Computing, STOC’95*, pages 596–605, New York, NY, USA, 1995. ACM.
  - 26 Satyanarayana V. Lokam. Spectral methods for matrix rigidity with applications to size–depth trade-offs and communication complexity. *Journal of Computer and System Sciences*, 63(3):449–473, 2001.
  - 27 Carsten Lund, Lance Fortnow, Howard Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. *J. ACM*, 39(4):859–868, October 1992.
  - 28 Michael Mitzenmacher and Eli Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, New York, NY, USA, 2005.

- 29 Ilan Newman and Mario Szegedy. Public vs. private coin flips in one round communication games (extended abstract). In Gary L. Miller, editor, *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, May 22-24, 1996*, pages 561–570. ACM, 1996.
- 30 Noam Nisan and Avi Wigderson. Rounds in communication complexity revisited. In *Proceedings of the Twenty-third Annual ACM Symposium on Theory of Computing, STOC'91*, pages 419–429, New York, NY, USA, 1991. ACM.
- 31 Y. Ofman. On the algorithmic complexity of discrete functions. *Doklady Akademii Nauk SSSR*, 145:48–51, 1962. English Translation in Soviet Physics Doklady 7: 589-591, 1963.
- 32 Charalampos Papamanthou, Elaine Shi, Roberto Tamassia, and Ke Yi. Streaming authenticated data structures. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology – EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, volume 7881 of *Lecture Notes in Computer Science*, pages 353–370. Springer, 2013.
- 33 Ran Raz. Quantum information and the PCP theorem. *Algorithmica*, 55(3):462–489, 2009.
- 34 Dominique Schröder and Heike Schröder. Verifiable data streaming. In Ting Yu, George Danezis, and Virgil D. Gligor, editors, *the ACM Conference on Computer and Communications Security, CCS'12, Raleigh, NC, USA, October 16-18, 2012*, pages 953–964. ACM, 2012.
- 35 Justin Thaler. Time-optimal interactive proofs for circuit evaluation. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013 – 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II*, volume 8043 of *Lecture Notes in Computer Science*, pages 71–89. Springer, 2013.
- 36 Victor Vu, Srinath T. V. Setty, Andrew J. Blumberg, and Michael Walfish. A hybrid architecture for interactive verifiable computation. In *2013 IEEE Symposium on Security and Privacy, SP 2013, Berkeley, CA, USA, May 19-22, 2013*, pages 223–237. IEEE Computer Society, 2013.
- 37 C.S. Wallace. A suggestion for a fast multiplier. *Electronic Computers, IEEE Transactions on*, EC-13(1):14–17, Feb 1964.

# Identifying an Honest $\text{EXP}^{\text{NP}}$ Oracle Among Many

Shuichi Hirahara

Department of Computer Science, The University of Tokyo  
7-3-1 Hongo, Bunkyo-ku, Tokyo 133-8654, Japan  
hirahara@is.s.u-tokyo.ac.jp

---

## Abstract

We provide a general framework to remove short advice by formulating the following computational task for a function  $f$ : given two oracles at least one of which is honest (*i.e.* correctly computes  $f$  on all inputs) as well as an input, the task is to compute  $f$  on the input with the help of the oracles by a probabilistic polynomial-time machine, which we shall call a *selector*. We characterize the languages for which short advice can be removed by the notion of selector: a paddable language has a selector if and only if short advice of a probabilistic machine that accepts the language can be removed under any relativized world.

Previously, instance checkers have served as a useful tool to remove short advice of probabilistic computation. We indicate that existence of instance checkers is a property stronger than that of removing short advice: although no instance checker for  $\text{EXP}^{\text{NP}}$ -complete languages exists unless  $\text{EXP}^{\text{NP}} = \text{NEXP}$ , we prove that there exists a selector for any  $\text{EXP}^{\text{NP}}$ -complete language, by building on the proof of  $\text{MIP} = \text{NEXP}$  by Babai, Fortnow, and Lund (1991).

**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** nonuniform complexity, short advice, instance checker, interactive proof systems, probabilistic checkable proofs

**Digital Object Identifier** 10.4230/LIPIcs.CCC.2015.244

## 1 Introduction

Blum and Kannan [10] introduced the notion of *instance checker*. Roughly speaking, an instance checker for a function  $f$  is an efficient probabilistic machine that, given access to an oracle, checks if the oracle computes  $f(x)$  correctly on a given instance  $x$ ; the oracle models a possibly buggy program that purports to compute  $f$ , and an instance checker verifies whether the program works correctly on a given instance.

The notion of instance checker is intimately related to interactive proof systems: the line of work showing the power of interactive proofs [22, 24, 6] yielded instance checkers for  $\text{P}^{\#P}$ -,  $\text{PSPACE}$ -, and  $\text{EXP}$ -complete languages; in addition, Blum and Kannan [10] gave a characterization of the languages with an instance checker by a function-restricted interactive proof system. Since any language with an interactive proof protocol is in  $\text{NEXP}$  [17], any language with an instance checker must be in  $\text{NEXP} \cap \text{coNEXP}$ .

In this paper, we investigate a computational task weaker than instance checking of a (Boolean) function  $f$ : we are given access to two oracles (instead of a single oracle) as well as an input  $x$ ; again, both of the oracles purport to compute  $f$ ; however, it is assumed that at least one of the two oracles is *honest*, *i.e.* computes  $f(q)$  correctly on all inputs  $q$ ; and the task is to compute  $f(x)$  with the help of the oracles in polynomial time. We shall call a probabilistic machine doing the task a (*probabilistic*) *selector* for  $f$ .

If the answers of oracles on the input  $x$  agree, then we have only to output the answer, which is surely correct by the assumption. Thus, the task of a selector is essentially to



© Shuichi Hirahara;  
licensed under Creative Commons License CC-BY  
30th Conference on Computational Complexity (CCC'15).

Editor: David Zuckerman; pp. 244–263



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

identify the honest oracle when two oracles disagree on  $x$  (*i.e.* one of the oracles asserts that  $f(x) = 0$ , whereas the other asserts that  $f(x) = 1$ ).

Our main result shows that there exists a selector for  $\text{EXP}^{\text{NP}}$ -complete languages. We also show that the notion of selector does not change even if there are one honest oracle and polynomially many dishonest oracles. Thus, these results can be encapsulated in the following phrase: “identifying an honest oracle among many is *strictly* weaker than instance checking unless  $\text{EXP}^{\text{NP}} = \text{NEXP}$ .”

Although the task is weaker than instance checking, a situation in which one may assume existence of an honest oracle naturally arises out of computation with advice: Suppose, for example, that a (paddable) language  $L$  is computed by a probabilistic machine  $M$  with advice of one bit. We regard  $M$  with advice 0 and 1 as two oracles  $A_0$  and  $A_1$ , respectively. By the definition of advice, either  $A_0$  or  $A_1$  is honest on all the inputs (of the same length). Thus, the advice of one bit can be removed if  $L$  has a selector. We can in fact remove advice of size  $O(\log n)$ , since a selector can identify an honest oracle among polynomially many oracles.

## 1.1 Removing Short Advice for Probabilistic Computation

In early work as to removing short advice for probabilistic computation, Trevisan and Vadhan [25] gave an insight into the potential of instance checkability: they demonstrated that instance checkability can be exploited to remove short advice. Based on the existence of an instance checker for  $\text{EXP}$ -complete languages, they showed a quantitative tradeoff from a uniform worst-case-hardness assumption (*i.e.*  $\text{EXP} \not\subseteq \text{BPTIME}(t(n^{O(1)}))$ ) to average-case hardness of  $\text{EXP}$  (*i.e.*  $\text{EXP}$  contains languages that cannot be solved by probabilistic computation on a fraction better than  $\frac{1}{2} + \frac{1}{t}$  of inputs in time  $t$ ).

They also argued that their result cannot be obtained via *black-box uniform reductions*. Typical constructions of a worst-case to average-case connection are based on the following scheme: we convert a function  $f$  into another function  $f'$ , which is an error-correcting code of  $f$ ; if we have a “black-box” algorithm that computes  $f'$  on a fraction greater than  $\frac{1}{2} + \epsilon$  of the inputs, then a probabilistic machine that takes advice can compute  $f$  on all inputs by decoding  $f'$ . Since it is impossible to uniquely decode  $f'$  for small  $\epsilon$ , the advice is used to identify  $f$  and is provably indispensable.

Indeed, it was the instance checkability of  $\text{EXP}$ -complete languages that broke the black-box construction in the proof of Trevisan and Vadhan; the instance checkability enabled them to remove advice of logarithmic size. Therefore, it will be helpful for future research to closely understand the property that they actually exploited.

Subsequent to their work, instance checkability has since been exploited to cope with short advice for probabilistic computation: for example, Barak [7] proved the first hierarchy theorem for probabilistic computation with short advice; Buhrman, Fortnow, and Santhanam [12] unconditionally separated  $\text{BPEXP}$  from  $\text{BPP}$  with advice of subpolynomial size; and Buhrman, Fortnow, Koucký, and Loff [11] gave some evidences that a deterministic efficient computation with oracle access to the set of Kolmogorov-random strings can be simulated by a probabilistic efficient computation.

## 1.2 Our Results

In fact, the notion of selector captures a property of removing short advice:

► **Theorem 1.1.** *Let  $L$  be an arbitrary paddable language. The following are equivalent:*

1. *There exists a selector for  $L$ .*
2. *For any oracle  $R \subseteq \{0, 1\}^*$ , it holds that  $L \in \text{BPP}^R // \log$  implies  $L \in \text{BPP}^R$ .*



That is, a paddable language has a selector if and only if short advice can be removed under any relativized world. (“//” means advice that can depend on coin flips of probabilistic machines as well as input length [25].)

In addition, we construct a selector for  $\text{EXP}^{\text{NP}}$ -complete languages, thereby indicating an essential difference between selectors and instance checkers. We also give an upper bound on the languages with a selector:

► **Theorem 1.2** (Main Theorem).

1. Every  $\text{EXP}^{\text{NP}}$ -complete language has a selector.
2. Any language with a selector is in  $\text{S}_2^{\text{EXP}}$  (which is an exponential-time analogue of  $\text{S}_2^{\text{P}}$ ).

Thus, existence of an instance checker is a property stronger than that of removing short advice (or, equivalently, existence of a selector): although no instance checker for  $\text{EXP}^{\text{NP}}$ -complete languages exists unless  $\text{EXP}^{\text{NP}} = \text{NEXP}$ , short advice of a probabilistic machine that accepts  $\text{EXP}^{\text{NP}}$ -complete languages can be removed.

## Our Techniques

The most technical part of this paper is a proof of the main theorem (Theorem 1.2, Part 1). In order to construct a selector for  $\text{EXP}^{\text{NP}}$ -complete languages, we build on the proof of  $\text{MIP} = \text{NEXP}$  by Babai, Fortnow, and Lund [6]. As pointed out by Gábor Tardos in the paper [6], the complexity of honest provers of the interactive proof system for  $\text{NEXP}$ -complete languages can be bounded above by  $\text{EXP}^{\text{NP}}$ . We crucially use this fact to check satisfiability of an exponential-sized formula with the help of an  $\text{EXP}^{\text{NP}}$ -complete oracle. We also compare two exponential-sized strings by performing a binary search.

Thanks to plenty of machinery that has been cultivated together with interactive proof systems, program checking, and PCPs, we can prove the main theorem by careful combinations of such machinery. For example, we exploit a multilinearity test [6] and the self-correction of low-degree polynomials [8, 21].

Due to the usage of arithmetization, we suspect that our proof of the main theorem does algebraize [1] but does not relativize.

## Variants of Selectors

We also investigate other variants of selectors: a *deterministic selector* and a *nonadaptive deterministic selector*. We focus on the “suprema” of the languages with a selector, namely, upper bounds on these languages and existence of a selector for languages complete for a complexity class that is close to the upper bounds. (Note that the languages with a selector are not necessarily closed downward. For example, although  $\text{NEXP} \subseteq \text{EXP}^{\text{NP}}$ , we do not know whether  $\text{NEXP}$ -complete languages have a selector or not.)

For a nonadaptive deterministic selector, we prove polynomial-time analogues of Theorem 1.2:

► **Theorem 1.3.**

1. Every  $\text{P}^{\text{NP}}$ -complete language has a nonadaptive deterministic selector.
2. Any language with a nonadaptive deterministic selector is in  $\text{S}_2^{\text{P}}$ .

The proofs of this theorem will clearly illustrate the basic ideas for Theorem 1.2.

Notice that  $\text{P}^{\text{NP}}$  is close to the upper bound  $\text{S}_2^{\text{P}}$  since  $\text{P}^{\text{NP}} \subseteq \text{S}_2^{\text{P}} \subseteq \text{ZPP}^{\text{NP}}$  [23, 14]. (Under suitable hardness assumptions, it holds that  $\text{P}^{\text{NP}} = \text{S}_2^{\text{P}}$  by derandomization [19].)

For a deterministic selector, the supremum is  $\text{PSPACE}$ :



► **Theorem 1.4.**

1. Every PSPACE-complete language has a deterministic selector. More generally, any downward self-reducible language has a deterministic selector.
2. Any language with a deterministic selector is in PSPACE.

As with Theorem 1.1, a property of removing short advice for deterministic computation can be characterized by existence of a deterministic selector:

► **Theorem 1.5.** *Let  $L$  be an arbitrary paddable language. The following are equivalent:*

1. *There exists a deterministic selector for  $L$ .*
2. *For any oracle  $R \subseteq \{0, 1\}^*$ , it holds that  $L \in P^R/\log$  implies  $L \in P^R$ .*

### 1.3 Comparison with Prior Work

In seminal work by Karp and Lipton [18] as to collapses of a uniform class contained in a nonuniform class, it was shown that  $NP \subseteq P/\log$  implies  $NP \subseteq P$  and  $PSPACE \subseteq P/\log$  implies  $PSPACE \subseteq P$ . These results are essentially equivalent to the existence of deterministic selectors for NP- and PSPACE-complete languages, respectively.

Fortnow and Klivans [16] observed that  $NEXP \subseteq BPP/\log$  implies  $NEXP = BPP$  by combining previous results. Similarly, it is folklore that  $EXP^{NP} \subseteq BPP/\log$  implies  $EXP^{NP} = BPP$ . This follows by combining the result by Buhrman and Homer [13] stating that  $EXP^{NP} \subseteq EXP/\text{poly}$  implies  $EXP^{NP} = EXP$ , the existence of an instance checker (or a selector) for EXP-complete languages, and  $BPP/\log \subseteq P/\text{poly}$  (see [16]).

We clarify the differences between the folklore and our results in two respects. First, our results can be relativized on the right-hand side. Second, selectors can be used to quantitatively remove advice of logarithmic size: if we allow a machine to run in time  $t$  (instead of polynomial time), then advice of size  $\log t$  can be removed.

► **Corollary 1.6** (Analogous to Proposition 5.6 in [25]). *There are an  $EXP^{NP}$ -complete language  $L$  and a constant  $d \in \mathbb{N}$  such that, for any nice time bound<sup>1</sup>  $t: \mathbb{N} \rightarrow \mathbb{N}$  and any oracle  $R \subseteq \{0, 1\}^*$ , if  $L \in BPTIME^R(t(n))/\log t(n)$  then  $L \in BPTIME^R(t(n^d))$ .*

We mention in passing that, by substituting selectors for instance checkers in the proofs of Trevisan and Vadhan [25], one can obtain a quantitative tradeoff from a uniform worst-case-hardness assumption on  $EXP^{NP}$  to a uniform average-case hardness of  $EXP^{NP}$  (see [25, Theorem 5.7]).

### 1.4 Application: Random Strings vs. Randomized Computation

In Section 6, we will give another application in order to demonstrate usefulness of the notion of selector, by simply substituting selectors for instance checkers in the previous work by Buhrman, Fortnow, Koucký, and Loff [11].

They tried to show that a deterministic polynomial-time computation with oracle access to the set of Kolmogorov-random strings is, in some sense, equivalent to a probabilistic polynomial-time computation; they modeled oracle access to the set of Kolmogorov-random strings as advice strings of high nonuniform complexity. Although the nonuniform complexity of the advice strings is required to be much higher than that of Kolmogorov-random strings, they showed, as a partial result, that if a language  $L$  can be solved in deterministic polynomial

<sup>1</sup> Although the definition of a nice time bound is the same as in [25], we note that the condition  $t(n) \leq 2^n$  is not needed here.

time with high nonuniform advice, then  $L$  is in BPP with advice of almost linear size [11, Theorem 13].

Because the goal is to show that  $L$  is in BPP *without* any advice, they further observed that one can dispense with the advice of almost linear size if there exists an instance checker for  $L$ . From this observation, they showed that, for any class  $\mathcal{C} \in \{\text{NP}, \text{P}^{\#\text{P}}, \text{PSPACE}, \text{EXP}\}$ , if some  $\mathcal{C}$ -complete language can be solved in deterministic polynomial time with high nonuniform advice, then  $\mathcal{C} \subseteq \text{BPP}$  [11, Theorem 15].

In fact, they proved this result by analyzing the two cases: For  $\mathcal{C} \in \{\text{P}^{\#\text{P}}, \text{PSPACE}, \text{EXP}\}$ , they used an instance checker for  $\mathcal{C}$ -complete languages, whose existence was shown by [22, 24, 6]; Unfortunately, because it is not known whether NP-complete languages have instance checkers or not, they needed to prove the result in another way solely for  $\mathcal{C} = \text{NP}$ .

The notion of selector, however, enables us to show the result in a unified way and to extend the result from  $\{\text{NP}, \text{P}^{\#\text{P}}, \text{PSPACE}, \text{EXP}\}$  to any classes whose complete languages have a selector. Given the fact that many languages have selectors (*e.g.* languages with instance checkers and downward self-reducible languages), it becomes more plausible that we can dispense with the advice of almost linear size; thereby we slightly strengthen the connection between Kolmogorov-random strings and randomized computation.

## Organization

In Section 2, we give formal definitions, common properties of selectors, and a proof of Theorem 1.1. Sections 3, 4, and 5 are devoted to investigating nonadaptive deterministic selectors, probabilistic selectors, and deterministic selectors, respectively. We mention some possible directions for future work in Section 7.

## Preliminaries and Notations

We assume that the reader is familiar with basics of computational complexity (*e.g.* [2]).

For a Turing machine  $M$ , let  $M(x)$  denote the output of  $M$  on input  $x \in \{0, 1\}^*$ . For an oracle Turing machine  $M$  and oracles  $A_0, A_1 \subseteq \{0, 1\}^*$ , let  $M^{A_0, A_1}$  represent a machine equipped with access to oracle  $A \subseteq \{0, 1\}^*$  such that  $A(i \cdot q) = A_i(q)$ , for each  $i \in \{0, 1\}$  and for any  $q \in \{0, 1\}^*$ . We identify false and true with 0 and 1, respectively. We also identify a language  $L \subseteq \{0, 1\}^*$  with its characteristic function from  $\{0, 1\}^*$  to  $\{0, 1\}$ . For a Boolean formula  $\varphi$  in  $n$  variables, we abuse notation and write  $\varphi: \{0, 1\}^n \rightarrow \{0, 1\}$ .

We say that a language  $L$  is paddable if there exists a polynomial-time machine that, on input  $(x, 1^m)$  where  $x \in \{0, 1\}^n$  and  $n \leq m$ , outputs a string  $y$  of length  $m$  such that  $y \in L$  if and only if  $x \in L$ .

## 2 Definitions and Common Properties of Selectors

In this section, we give formal definitions of selectors and show common properties that all types of selectors have. First, we define a probabilistic selector:

► **Definition 2.1** (Probabilistic Selector). A (*probabilistic*) *selector*  $S$  for a language  $L \subseteq \{0, 1\}^*$  is a probabilistic polynomial-time oracle Turing machine which computes  $L$  with high probability, given arbitrary two oracles  $A_0, A_1 \subseteq \{0, 1\}^*$  such that  $A_0$  or  $A_1$  is equal to  $L$ . That is, for any input  $x \in \{0, 1\}^*$  and oracles  $A_0, A_1 \subseteq \{0, 1\}^*$ ,

$$L \in \{A_0, A_1\} \implies \Pr[S^{A_0, A_1}(x) = L(x)] \geq \frac{2}{3},$$

where the probability is taken over coin flips of  $S$ .

Note that the success probability  $\frac{2}{3}$  in Definition 2.1 can be enhanced by repetitions. We often abbreviate a probabilistic selector as a selector.

An oracle equal to  $L$  is said to be *honest*; otherwise it is said to be *dishonest*.

Next, we define a deterministic selector and a nonadaptive deterministic selector:

► **Definition 2.2** (Deterministic Selector). A *deterministic selector* for a language  $L$  is a deterministic polynomial-time oracle machine  $S$  such that  $S^{L,X}(x) = S^{X,L}(x) = L(x)$  for any oracle  $X \subseteq \{0, 1\}^*$  and for any input  $x \in \{0, 1\}^*$ .

► **Definition 2.3** (Nonadaptive Deterministic Selector). A *nonadaptive deterministic selector*  $S$  for a language  $L$  is a deterministic polynomial-time oracle machine such that

- $S^{L,X}(x) = S^{X,L}(x) = L(x)$  for any oracle  $X \subseteq \{0, 1\}^*$  and any input  $x \in \{0, 1\}^*$ , and
- $S$  is nonadaptive, *i.e.* there exists a polynomial-time machine which, on input  $x \in \{0, 1\}^*$ , outputs the query set  $Q(x)$  of all the queries that  $S$  makes to either of the oracles.

We state a useful structural property:

► **Proposition 2.4.** *The class of the languages with a selector is closed under polynomial-time Turing equivalence. Namely,  $L_1 \leq_T^P L_2$  and  $L_2 \leq_T^P L_1$  imply that if  $L_1$  has a selector then so does  $L_2$ .*

*In particular, it is closed under complement. Moreover, for any complexity class  $\mathcal{C}$ , if a specific  $\mathcal{C}$ -complete language has a selector, then so does an arbitrary  $\mathcal{C}$ -complete language.*

**Proof.** The proof is essentially the same with Beigel's theorem [10], which shows the same closure property of instance checkers. The idea is as follows: reduce a  $L_2$  problem to a  $L_1$  problem by using the reducibility from  $L_2$  to  $L_1$ , and solve the  $L_1$  problem by running a selector for  $L_1$ , while converting its query (which is an instance of  $L_1$ ) into an instance of  $L_2$ .

Let  $M_{ij}$  be a polynomial-time oracle machine that witnesses the polynomial-time Turing reduction  $L_i \leq_T^P L_j$  for each  $(i, j) \in \{(1, 2), (2, 1)\}$  (that is,  $M_{ij}^{L_j}(x) = L_i(x)$  for any  $x$ ), and  $S$  be a selector for  $L_1$ . The following algorithm yields a selector for  $L_2$ : Given an input  $x \in \{0, 1\}^n$  and two oracles  $A_0, A_1$ , simulate  $M_{21}(x)$  in order to compute  $L_2(x)$ . If  $M_{21}$  makes a query  $q$ , then we try to answer it with  $L_1(q)$ , by running  $S(q)$ . If  $S$  makes a query  $q'$  to the  $i$ th oracle ( $i \in \{0, 1\}$ ), then answer it with  $M_{12}^{A_i}(q')$ .

Let  $A_i$  be an honest oracle (*i.e.*  $A_i = L_2$ ). Then, we have  $M_{12}^{A_i}(q') = M_{12}^{L_2}(q') = L_1(q')$ , and hence  $S(q)$  is simulated under the existence of the honest oracle; thus it outputs  $L_1(q)$  correctly with high probability (say, with probability at least  $1 - 2^{-n}$ , by running the selector  $O(n)$  times). Therefore, the simulation of  $M_{21}(x)$  results in outputting  $L_2(x)$  with probability at least  $1 - 2^{-n}n^{O(1)}$ . ◀

► **Remark 2.5.** Similarly, the class of languages with a deterministic selector is closed under polynomial-time Turing equivalence, and the class of languages with a nonadaptive deterministic selector is closed under polynomial-time truth-table (*i.e.* nonadaptive) equivalence.

To prove Theorem 1.1, we show that the definitions of selectors are robust even if we consider a situation in which we are given polynomially many oracles.

► **Lemma 2.6.** *For any language  $L \subseteq \{0, 1\}^*$ , the following are equivalent:*

1. *There exists a selector for  $L$ .*
2. *There exists a selector for  $L$  that identifies an honest oracle among polynomially many oracles.*

The latter can be formally stated as follows: for any polynomial  $m: \mathbb{N} \rightarrow \mathbb{N}$ , there exists a probabilistic polynomial-time oracle Turing machine  $S$  such that, on input length  $n \in \mathbb{N}$ , it holds that  $\Pr [S^A(x) = L(x)] \geq \frac{2}{3}$  for any  $x \in \{0, 1\}^n$ , where  $A$  is an arbitrary oracle such that there exists an index  $i \in \{1, \dots, m(n)\}$  that satisfies  $A(i, q) = L(q)$  for all  $q \in \{0, 1\}^*$ .

**Proof.** The one direction is obvious: If there exists a selector that works among  $m(n)$  oracles, then letting  $m(n) := 2$  yields a selector that works among two oracles.

Conversely, let  $S$  be a selector (that identifies an honest oracle among two oracles) with probability at least  $1 - \frac{1}{3m(n)}$ . Given an oracle  $A$ , let  $A_i(q)$  denote  $A(i, q)$  for any  $i \in \mathbb{N}$ . On input  $x \in \{0, 1\}^n$ , we first make a query  $x$  to all the oracles  $A_1, \dots, A_{m(n)}$ , and divide them into the two sets according to their answers:

$$C_0 = \{j \in \{1, \dots, m(n)\} \mid A_j(x) = 0\},$$

$$C_1 = \{k \in \{1, \dots, m(n)\} \mid A_k(x) = 1\}.$$

That is,  $C_\alpha$  ( $\alpha \in \{0, 1\}$ ) is the set of the indices of all the oracles asserting that  $L(x) = \alpha$ .

Next, we repeat the following until  $C_0 = \emptyset$  or  $C_1 = \emptyset$ : Pick arbitrary elements  $j \in C_0$  and  $k \in C_1$ . We check which is a supposedly honest oracle by running  $S^{A_j, A_k}$  on input  $x$ . If  $S^{A_j, A_k}(x) = 0$ , then we doubt  $A_k$  and thus eliminate  $k$  from  $C_1$ ; Otherwise we doubt  $A_j$  and eliminate  $j$  from  $C_0$ .

Finally, we output 1 if and only if  $C_1 \neq \emptyset$ .

Now let us analyze this algorithm. It runs in polynomial time because  $|C_0| + |C_1|$  is decreased by one in each repetition.

We claim the correctness of the algorithm. For simplicity, we assume that  $L(x) = 0$ . Then, there exists an index  $i \in \{1, \dots, m(n)\}$  such that  $A_i$  is honest and  $i \in C_0$ . If  $i \in C_0$  and some  $k \in C_1$  are picked in a repetition, then  $\Pr [S^{A_i, A_k}(x) = 0] \geq 1 - \frac{1}{3m(n)}$ . That is,  $i$  remains in  $C_0$  with probability at least  $1 - \frac{1}{3m(n)}$ . Since  $i$  is picked at most  $|C_1|$  ( $\leq m(n)$ ) times, the probability that  $i$  remains in  $C_0$  is at least  $1 - m(n) \cdot \frac{1}{3m(n)} = \frac{2}{3}$ .  $\blacktriangleleft$

► **Remark 2.7.** Although Lemma 2.6 is stated only for a probabilistic selector, analogous statements hold for a deterministic selector and a nonadaptive deterministic selector. For a deterministic selector, one can easily check that the same proof works. For a nonadaptive deterministic selector, we must compute the query set in polynomial time. On input  $x$ , let  $Q(x)$  denote the query (to either  $A_0$  or  $A_1$ ) set of a selector that identifies an honest oracle among two oracles. Then we can define all the set of possible queries as  $Q'(x) := \{(i, q) \in \mathbb{N} \times \{0, 1\}^* \mid 1 \leq i \leq m(|x|), q \in Q(x) \cup \{x\}\}$ , which is clearly computable in polynomial time.

By using Lemma 2.6, we characterize the class of the paddable languages with a selector by the property that short advice can be removed under any relativized world. In fact, we can prove a statement stronger than Theorem 1.1:

► **Theorem 2.8.**

1. For any paddable language  $L$ , if  $L$  has a selector, then  $L \in \text{BPP}^R // \log$  implies  $L \in \text{BPP}^R$  for any oracle  $R \subseteq \{0, 1\}^*$ .
2. For any language  $L$ , if  $L \in \text{P}^R / 1$  implies  $L \in \text{BPP}^R$  for any oracle  $R \subseteq \{0, 1\}^*$ , then  $L$  has a selector.

As a corollary, we immediately obtain Theorem 1.1 (note that  $\text{P}^R / 1 \subseteq \text{BPP}^R // \log$ ).

**Proof.**

**Part 1.** Let  $M$  be a polynomial-time oracle machine which witnesses  $L \in \text{BPP}^R // a$ , where  $a(n) = O(\log n)$ . That is, there exists an advice function  $\alpha: \{0, 1\}^* \rightarrow \{0, 1\}^*$  such that, for every  $n \in \mathbb{N}$ ,

$$\Pr_{r \in \{0, 1\}^{t(n)}} [\forall x \in \{0, 1\}^n, M^R(x, r, \alpha(r)) = L(x)] \geq \frac{5}{6}, \quad (1)$$

where  $|\alpha(r)| = a(n)$  and  $t$  is a polynomial (see also [25, Definition 5.1]).

Let  $l(n)$  ( $= n^{O(1)}$ ) be an upper bound on the running time of a selector for  $L$  on inputs of length  $n$ . By Lemma 2.6, there exists a selector  $S$  that can identify an honest oracle among  $m(n)$  oracles for  $m(n) := 2^{a(l(n))} = n^{O(1)}$  with probability at least  $\frac{5}{6}$ . By padding, we may assume that  $S$  makes only queries of length exactly  $l(n)$  on each input length  $n \in \mathbb{N}$ .

Consider the following probabilistic algorithm: On input  $x \in \{0, 1\}^n$ , pick a string  $r \in_R \{0, 1\}^{t(l(n))}$  uniformly at random, and define oracles by  $A_i(q) := M^R(q, r, i)$  for any  $q \in \{0, 1\}^{l(n)}$ , where  $i \in \{1, \dots, m(n)\}$  is identified with  $i \in \{0, 1\}^{a(l(n))}$ . Simulate  $S$  on input  $x$ , answering its queries  $q \in \{0, 1\}^{l(n)}$  to  $A_i$  by computing  $M^R(q, r, i)$ .

If a “good” string  $r$  is picked (whose probability is at least  $\frac{5}{6}$  by (1)), then we have  $A_i(q) = M^R(q, r, i) = L(q)$  for any  $q \in \{0, 1\}^{l(n)}$ , where  $i = \alpha(r)$ . That is,  $A_i$  is honest for some  $i$  with probability at least  $\frac{5}{6}$ . Thus, the algorithm computes  $L$  correctly with probability at least  $1 - \frac{1}{6} - \frac{1}{6} = \frac{2}{3}$ .

**Part 2.** We prove the contraposition. Assume that  $L$  does not have any selectors.

Recall that we regard the computation given oracle access to two oracles  $R_0, R_1$ , namely  $M^{R_0, R_1}$ , as  $M^R$  where  $R(i \cdot q) = R_i(q)$  for each  $i \in \{0, 1\}$ . Thus, the goal is to show that there exist oracles  $R_0, R_1 \subseteq \{0, 1\}^*$  such that  $L \in \text{P}^{R_0, R_1} / 1$  and  $L \notin \text{BPP}^{R_0, R_1}$ .

We use a diagonalization argument on all the probabilistic polynomial-time oracle machine  $M_1, M_2, \dots$ . We construct  $R_0^{(e)}, R_1^{(e)}$  at stage  $e \in \mathbb{N}$ , and then define  $R_i := \bigcup_e R_i^{(e)}$  for each  $i \in \{0, 1\}$ .

We will construct them so that, for each  $n \in \mathbb{N}$ , there exists  $j_n \in \{0, 1\}$  such that  $R_{j_n}(q) = L(q)$  for any  $q \in \{0, 1\}^n$ . Thus,  $L \in \text{P}^{R_0, R_1} / 1$  holds because we can make a query  $x$  to obtain  $R_{j_n}(x) = L(x)$  with advice  $\{j_n\}_{n \in \mathbb{N}}$  of one bit.

Let us now construct  $R_0^{(e)}, R_1^{(e)}$ , and  $l^{(e)} \in \mathbb{Z}$ , where  $l^{(e)}$  represents the maximum length of the strings that have been fixed. At stage  $e = 0$ , we set  $R_0^{(0)} = R_1^{(0)} = \emptyset$ , and  $l^{(0)} := -1$ .

At stage  $e \geq 1$ , we claim that  $R_0^{(e-1)}$  and  $R_1^{(e-1)}$  can be extended so that some input  $x^{(e)}$  can fool  $M_e$ :

► **Claim 2.9.** For each  $e \geq 1$ , there exist oracles  $A_0, A_1 \subseteq \{0, 1\}^*$  and a string  $x^{(e)} \in \{0, 1\}^*$  such that

1.  $A_i$  agrees with  $R_i^{(e-1)}$  on all the strings of length at most  $l^{(e-1)}$  for each  $i \in \{0, 1\}$ ,
2. either  $A_0$  or  $A_1$  agrees with  $L$  on all the strings of length greater than  $l^{(e-1)}$ , and
3.  $\Pr [M_e^{A_0, A_1}(x^{(e)}) = L(x^{(e)})] < \frac{2}{3}$ .

**Proof of Claim 2.9.** Assume otherwise. That is, for any oracles  $A_0, A_1 \subseteq \{0, 1\}^*$  and string  $x \in \{0, 1\}^*$ , we have  $\Pr [M_e^{A_0, A_1}(x) = L(x)] \geq \frac{2}{3}$  if Properties 1 and 2 hold. Then, the following algorithm yields a selector for  $L$ , which contradicts the assumption: we hardwire all the strings in  $R_i^{(e-1)}$  of length at most  $l^{(e-1)}$  into a table; given oracles  $A_0, A_1$  one of which agrees with  $L$ , we simulate  $M_e$ , answering its queries  $q$  to  $A_i$  ( $i \in \{0, 1\}$ ) with the content of the table if  $|q| \leq l^{(e-1)}$  and with  $A_i(q)$  otherwise. ◀

Define  $l^{(e)}$  ( $> l^{(e-1)}$ ) as an upper bound on the length of the queries that  $M_e^{A_0, A_1}(x^{(e)})$  makes. Then, define  $R_i^{(e)}$  as  $R_i^{(e)}(q) := R_i^{(e-1)}(q) = A_i(q)$  if  $|q| \leq l^{(e-1)}$ ;  $R_i^{(e)}(q) := A_i(q)$  if  $l^{(e-1)} < |q| \leq l^{(e)}$ ; and  $R_i^{(e)}(q) = 0$  otherwise, for each  $q \in \{0, 1\}^*$ . This completes the construction of stage  $e$ .

On one hand,  $x^{(e)}$  witnesses  $M_e^{R_0, R_1}$  not computing  $L$  on input  $x^{(e)}$  for any  $e \geq 1$ , by Property 3; thus, we have  $L \notin \text{BPP}^{R_0, R_1}$ . On the other hand, for each input length  $n \in \mathbb{N}$ , either  $R_0$  or  $R_1$  agrees with  $L$  on  $\{0, 1\}^n$ , by Property 2; thus, we have  $L \in \text{P}^{R_0, R_1}/1$ . ◀

► **Remark 2.10.** Again, the analogous statement (Theorem 1.5) holds for a deterministic selector. A proof is essentially the same and hence is omitted.

One can also prove the quantitative version (Corollary 1.6) of Part 1 of Theorem 2.8 by changing parameters in the proofs of Theorem 2.8 and Lemma 2.6.

### 3 Nonadaptive Deterministic Selector

In this section we prove Theorem 1.3.

We first prove Part 1 of Theorem 1.3, which states that *every*  $\text{P}^{\text{NP}}$ -complete language has a nonadaptive deterministic selector. It is sufficient to show that a *specific*  $\text{P}^{\text{NP}}$ -complete language has a selector (recall Proposition 2.4 and Remark 2.5). We construct a nonadaptive deterministic selector for the following canonical  $\text{P}^{\text{NP}}$ -complete language (see [20] for a proof of its completeness).

► **Definition 3.1** (Lexicographically Maximum Satisfying Assignment; Krentel [20]). The *lexicographically maximum satisfying assignment problem* contains all the pairs  $(\varphi, k)$  such that  $\varphi: \{0, 1\}^n \rightarrow \{0, 1\}$  is a satisfiable Boolean formula in  $n$  variables for some  $n \in \mathbb{N}$ , and  $a_k = 1$ , where  $a_1 \cdots a_n \in \{0, 1\}^n$  denotes the lexicographically maximum satisfying assignment of  $\varphi$ .

In other words, the lexicographically maximum satisfying assignment problem is the decision version of the problem of answering, given a Boolean formula  $\varphi$  in  $n$  variables, the lexicographically maximum satisfying assignment if  $\varphi$  is satisfiable and  $0^n$  otherwise. Note that it is implicit in the definition that the answer is  $0^n$  for an unsatisfiable Boolean formula.

**Proof of Part 1 of Theorem 1.3.** We show an algorithm of a selector for the lexicographically maximum satisfying problem, together with its analysis. Let us call two oracles  $A_0$  and  $A_1$ .

On input  $(\varphi, k)$ , the set of all the queries that we make is  $\{(\varphi, j) \mid j \in \{1, \dots, n\}\}$ , where  $n \in \mathbb{N}$  is the number of variables in  $\varphi$ . The (presumably) lexicographically maximum satisfying assignment asserted by each oracle  $A_i$  ( $i \in \{0, 1\}$ ) can be obtained by concatenating the answers of the oracle, namely  $A_i(\varphi, 1) \cdot A_i(\varphi, 2) \cdots A_i(\varphi, n) =: v_i \in \{0, 1\}^n$ .

If the  $k$ th bits of  $v_0$  and  $v_1$  agree, then we simply output it because the oracles agree on input  $(\varphi, k)$ .

Otherwise  $v_0$  is not equal to  $v_1$ . Therefore, we may assume without loss of generality that  $v_0 < v_1$ . We check whether  $v_1$  is a satisfying assignment or not by evaluating  $\varphi(v_1)$ . If  $\varphi(v_1) = 1$ , then we trust the oracle  $A_1$  and output  $A_1(\varphi, k)$  because  $A_1$  showed a satisfying assignment larger than  $v_0$ ; otherwise we doubt  $A_1$  and output  $A_0(\varphi, k)$  because  $A_1$  tried to cheat us by answering an unsatisfying assignment. ◀

Then we show that any language with a nonadaptive deterministic selector is in  $\text{S}_2^{\text{P}}$ .

**Proof of Part 2 of Theorem 1.3.** Let  $L$  be a language with a nonadaptive deterministic selector  $S$ . We claim that  $L$  is in  $S_2^P$ . Let  $Q(x) = \{q_1, \dots, q_m\}$  be the query set of  $S$  on input  $x \in \{0, 1\}^*$ .

We consider the following polynomial-time machine  $M$ : Suppose that the input to  $M$  is  $(x, y, z) \in \{0, 1\}^n \times \{0, 1\}^m \times \{0, 1\}^m$ . Let  $y = y_1 \cdots y_m$  and  $z = z_1 \cdots z_m$ .  $M$  simulates the selector  $S$  on input  $x$ . If  $S$  makes a query  $q_i$  to the oracle  $A_0$ , then it is answered with  $y_i$ . Similarly, if  $S$  makes a query  $q_i$  to the oracle  $A_1$ , then it is answered with  $z_i$ .

Then, there exists  $y \in \{0, 1\}^m$  such that  $M(x, y, z) = L(x)$  for any  $z \in \{0, 1\}^m$ . Indeed, if  $y$  is the concatenation of  $L(q_1), \dots, L(q_m)$ , then by the definition of a nonadaptive deterministic selector,  $M(x, y, z)$  correctly outputs  $L(x)$  for any  $z \in \{0, 1\}^m$ , because all the queries that  $S$  makes to  $A_0$  are answered correctly. Similarly, there exists  $z \in \{0, 1\}^m$  such that  $M(x, y, z) = L(x)$  for any  $y \in \{0, 1\}^m$ . ◀

## 4 Probabilistic Selector

In this section we investigate probabilistic selectors.

First, we show that probabilistic selectors can be constructed based on instance checkers. An instance checker is formally defined as follows:

► **Definition 4.1** (Instance Checker [10]). An *instance checker*  $C$  for a language  $L$  is a probabilistic polynomial-time oracle machine such that, given any oracle  $A \subseteq \{0, 1\}^*$ ,

1. if  $A = L$  then  $C^A$  accepts with high probability, *i.e.*  $\Pr [C^A(x) = 1] \geq \frac{2}{3}$  on all the input  $x \in \{0, 1\}^*$ , and
2. for any input  $x \in \{0, 1\}^*$ , if  $A(x) \neq L(x)$  then  $C^A(x)$  rejects with high probability, *i.e.*  $\Pr [C^A(x) = 0] \geq \frac{2}{3}$ ,

where the probability is taken over coin flips of  $C$ .

► **Proposition 4.2.** *Every language with an instance checker has a selector.*

**Proof.** Suppose that a language  $L$  has an instance checker  $C$ . Given input  $x \in \{0, 1\}^*$  and two oracles  $A_0, A_1 \subseteq \{0, 1\}^*$ , we check which is honest,  $A_0$  or  $A_1$ , by computing  $C^{A_0}(x)$ . If  $C^{A_0}(x)$  accepts, then we trust  $A_0$  and output  $A_0(x)$ ; otherwise we doubt  $A_0$  and output  $A_1(x)$ .

Let us analyze the algorithm above. If  $A_0 = L$ , then  $C^{A_0}(x)$  accepts with probability at least  $\frac{2}{3}$ , and hence we can output  $A_0(x) = L(x)$  correctly with probability at least  $\frac{2}{3}$ .

Otherwise, it must hold that  $A_1 = L$ . If  $A_0(x) = L(x)$ , then we can surely output  $L(x)$  correctly since  $A_0(x) = A_1(x) = L(x)$ . If  $A_0(x) \neq L(x)$ , then  $C^{A_0}(x)$  rejects with probability at least  $\frac{2}{3}$ , and thus we can output  $A_1(x) = L(x)$  correctly with probability at least  $\frac{2}{3}$ . ◀

Next, we show an upper bound on the languages with a probabilistic selector. For completeness, we include a definition of  $S_2^{\text{exp}}$ , which is a straightforward exponential-time analogue of  $S_2^P$ :

► **Definition 4.3.** We say that a language  $L$  is in  $S_2^{\text{exp}}$  if there exist a time-constructible function  $t(n) = 2^{n^{O(1)}}$  and a Turing machine  $M$  running in time  $2^{|x|^{O(1)}}$  on input  $(x, \cdot, \cdot)$  such that, for any input  $x \in \{0, 1\}^*$ ,

$$\exists y \in \{0, 1\}^{t(|x|)}, \forall z \in \{0, 1\}^{t(|x|)}, M(x, y, z) = L(x),$$

$$\exists z \in \{0, 1\}^{t(|x|)}, \forall y \in \{0, 1\}^{t(|x|)}, M(x, y, z) = L(x).$$

The proof itself is essentially a corollary of Part 2 of Theorem 1.3:



**Proof of Part 2 of Theorem 1.2.** Notice that a probabilistic selector can be simulated by an exponential-time nonadaptive deterministic selector. In addition, every language with an exponential-time nonadaptive deterministic selector is in  $\text{S}_2^{\text{exp}}$ , which is an exponential-time analogue of Part 2 of Theorem 1.3. Combining these two facts, it follows that every language with a probabilistic selector is in  $\text{S}_2^{\text{exp}}$ . ◀

#### 4.1 Selector for $\text{EXP}^{\text{NP}}$ -complete Languages

In this subsection we prove the main theorem (Theorem 1.2, Part 1). That is, we construct a selector for  $\text{EXP}^{\text{NP}}$ -complete languages.

##### Proof Sketch

We sketch the proof of the main theorem. We will construct a selector for a specific  $\text{EXP}^{\text{NP}}$ -complete language, which is a problem of finding the lexicographically maximum satisfying assignment of a succinctly described Boolean formula  $F_\Phi: \{0, 1\}^{2^n} \rightarrow \{0, 1\}$ . The basic strategy to construct a selector for this language is the same with that of Part 1 of Theorem 1.3: Given access to two oracles  $A_0, A_1 \subseteq \{0, 1\}^*$ , we request them to reveal the presumably lexicographically maximum satisfying assignments  $V_0, V_1 \in \{0, 1\}^{2^n}$  asserted by  $A_0, A_1$ , respectively. The rest of the algorithm consists of two parts: First, we determine the larger assignment of  $V_0$  and  $V_1$ , checking whether  $V_0 < V_1$  or  $V_0 > V_1$ . Second, we verify whether the larger assignment satisfies the formula  $F_\Phi$  or not. Obviously, the obstacle is that there can be exponentially many variables and clauses in  $F_\Phi$ .

For the second part, Babai, Fortnow, and Lund [6] showed that, given access to provers (or, equivalently, an oracle), one can efficiently check that exponentially many constraints in  $F_\Phi$  are satisfied: basically, by encoding an assignment as a multilinear function and using arithmetization, it holds that the assignment satisfies all the clauses in  $F_\Phi$  if and only if the sum of some low-degree polynomials (that can be computed by the multilinear function and the arithmetization) over a subdomain  $\{0, 1\}^l$  is equal to 0, and the latter can be verified by using the sum-check protocol [22] (called the LFKN protocol in [6]). As pointed out by Gábor Tardos [6], since  $\text{EXP}^{\text{NP}}$  is capable of finding a satisfying assignment of an exponential-sized Boolean formula, the honest oracle in the protocol above can be implemented in  $\text{EXP}^{\text{NP}}$ ; thus, given access to an honest  $\text{EXP}^{\text{NP}}$ -complete oracle (which is  $A_0$  or  $A_1$ ), one can verify the satisfiability.

For the first part, we perform a binary search to obtain the lexicographically first index  $z$  such that  $V_0$  and  $V_1$  disagree. Thus, we need

1. to check if  $V_0 = V_1$  on some range of indices, and
2. to split the range into two parts.

We observe that these can be done if we encode a satisfying assignment by the multilinear extension (as with [6]): Let  $\mathbb{F}$  be a finite field. We regard the assignments  $V_0, V_1 \in \{0, 1\}^{2^n}$  as vectors in  $\mathbb{F}^{2^n}$ . There is a bijective correspondence between a vector  $V \in \mathbb{F}^{2^n}$  and a multilinear function  $\tilde{V}: \mathbb{F}^n \rightarrow \mathbb{F}$ . For example, if  $n = 2$  and  $V = (V_{00}, V_{01}, V_{10}, V_{11})$ , then

$$\tilde{V}(x_1, x_2) = V_{00}(1 - x_1)(1 - x_2) + V_{01}(1 - x_1)x_2 + V_{10}x_1(1 - x_2) + V_{11}x_1x_2.$$

For Part 1, we can rely on the polynomial identity testing: indeed, since the multilinear extension is bijective, we have  $V_0 \neq V_1$  if and only if these multilinear extensions  $\tilde{V}_0$  and  $\tilde{V}_1$  differ; thus, it is sufficient to check if the two low-degree polynomials  $\tilde{V}_0$  and  $\tilde{V}_1$  differ.

It is well known that, given access to two low-degree polynomials, one can efficiently check if these polynomials differ: given access to two functions  $\tilde{V}_0, \tilde{V}_1$ , pick a random point



$u \in_R \mathbb{F}^n$  and check if  $\tilde{V}_0(u) \neq \tilde{V}_1(u)$ . Assuming that the functions are low-degree (which is true if they are multilinear), the Schwartz-Zippel lemma assures that  $\tilde{V}_0$  and  $\tilde{V}_1$  disagree on a large fraction of inputs if  $\tilde{V}_0 \neq \tilde{V}_1$ . Although it is possible that a dishonest oracle tries to cheat us by storing a high-degree polynomial, we can check whether or not the function stored by an oracle is close to some multilinear function, by using the multilinearity test [6].

For Part 2, we use the following simple fact: Fixing the first variable of a multilinear extension  $\tilde{V}$  to 0 or 1, we obtain multilinear extensions that correspond to the first or second part of  $V$ . In the example above, we obtain two multilinear functions:

$$\tilde{V}(0, x_2) = V_{00}(1 - x_2) + V_{01}x_2, \quad \tilde{V}(1, x_2) = V_{10}(1 - x_2) + V_{11}x_2.$$

These correspond to multilinear extensions of  $(V_{00}, V_{01})$  and  $(V_{10}, V_{11})$ , respectively, for  $n = 1$ . Thus, we can recursively compute the lexicographically first disagreement.

## Proof of the Main Theorem

Now we move on to the proof of the main theorem. We construct a selector for the following  $\text{EXP}^{\text{NP}}$ -complete language, which is an analogue of the NEXP-complete languages called the oracle-3-satisfiability problem in [6].

► **Definition 4.4** (Lexicographically Maximum Oracle-3-satisfying Assignment). Let  $m, n$  be nonnegative integers, and  $\Phi: \{0, 1\}^{m+3n+3} \rightarrow \{0, 1\}$  be a Boolean formula. For a Boolean function  $X: \{0, 1\}^n \rightarrow \{0, 1\}$ , define  $F_\Phi(X)$  as the following Boolean formula:

$$\bigwedge_{w \in \{0, 1\}^{m+3n}} \Phi(w, X(b_1), X(b_2), X(b_3)),$$

where  $w = (y, (b_1, b_2, b_3)) \in \{0, 1\}^m \times (\{0, 1\}^n)^3$ . A Boolean function  $X: \{0, 1\}^n \rightarrow \{0, 1\}$  is said to be an assignment of  $F_\Phi$ . For assignments  $X, Y: \{0, 1\}^n \rightarrow \{0, 1\}$ , we introduce the lexicographical ordering:  $X$  is less than  $Y$  if there exists an index  $b \in \{0, 1\}^n$  such that  $X(b) < Y(b)$  and  $X(b') = Y(b')$  for any  $b' < b$ . Let  $V_\Phi: \{0, 1\}^n \rightarrow \{0, 1\}$  denote the lexicographically maximum assignment such that  $F_\Phi(V_\Phi) = 1$  (i.e. the lexicographically maximum satisfying assignment of  $F_\Phi$ ); if there is no satisfying assignment, then define  $V_\Phi(b) = 0$  for any  $b \in \{0, 1\}^n$ .

The *lexicographically maximum oracle-3-satisfying assignment* is a problem of answering  $V_\Phi(b_{\text{in}})$ , given nonnegative integers  $m, n$ , a Boolean formula  $\Phi: \{0, 1\}^{m+3n+3} \rightarrow \{0, 1\}$ , and an index  $b_{\text{in}} \in \{0, 1\}^n$  as input.

We omit a proof of  $\text{EXP}^{\text{NP}}$ -completeness because this is a simple exponential-time analogue of the lexicographically maximum satisfying assignment language [20] (see also [6]).

Suppose that the input is a Boolean formula  $\Phi: \{0, 1\}^{m+3n+3} \rightarrow \{0, 1\}$  and an index  $b_{\text{in}}$ , and that we have access to two oracles  $A_0$  and  $A_1$ , one of which is honest.

## Encoding Assignments by the Multilinear Extension

As with the proof of  $\text{MIP} = \text{NEXP}$  [6], we encode a satisfying assignment by the multilinear extension. Let  $\mathbb{F}$  be a prime field such that  $|\mathbb{F}|$  is sufficiently large (but is bounded by a polynomial in the input size). We regard  $\{0, 1\} \subseteq \mathbb{F}$  in the canonical way. We say that a function  $f: \mathbb{F}^n \rightarrow \mathbb{F}$  is multilinear if it is a polynomial of degree at most 1 in each variable.

► **Proposition 4.5** (Multilinear Extension). *Let  $f: \{0, 1\}^n \rightarrow \mathbb{F}$  be an arbitrary function. Then, there exists a unique multilinear function  $\tilde{f}: \mathbb{F}^n \rightarrow \mathbb{F}$  such that  $f$  and  $\tilde{f}$  agree on  $\{0, 1\}^n$ .*

**Proof Sketch.** For a complete proof, the reader is referred to [6, Proposition 4.4]. Here, we note that the extension  $\tilde{f}$  can be explicitly written as

$$\tilde{f}(x) = \sum_{b \in \{0,1\}^n} f(b) \prod_{i=1}^n ((1-x_i)(1-b_i) + x_i b_i), \quad (2)$$

where  $b = (b_1, \dots, b_n)$  and  $x = (x_1, \dots, x_n) \in \mathbb{F}^n$ .  $\blacktriangleleft$

For the lexicographically maximum satisfying assignment  $V_\Phi: \{0,1\}^n \rightarrow \{0,1\} \subseteq \mathbb{F}$ , let  $\tilde{V}_\Phi: \mathbb{F}^n \rightarrow \mathbb{F}$  denote its multilinear extension.

We request the oracles to grant local access to  $\tilde{V}_\Phi$ . Formally, we consider the following search problem: given a Boolean formula  $\Phi$ , a prime  $|\mathbb{F}|$ , and  $x \in \mathbb{F}^n$ , the task is to output the value  $\tilde{V}_\Phi(x)$ . We regard this problem as a decision problem in the standard way. (Specifically, given the inputs specified above and auxiliary inputs  $k \in \mathbb{N}$  and  $b \in \{0,1\}$ , the task is to output one bit saying whether or not the  $k$ th bit of a binary representation of  $\tilde{V}_\Phi(x)$  is  $b$ .) The problem is still solvable in  $\text{EXP}^{\text{NP}}$ , by first computing  $V_\Phi$  in  $\text{EXP}^{\text{NP}}$  and then computing the expression (2) straightforwardly in exponential time.

Therefore, the problem can be reduced to the original  $\text{EXP}^{\text{NP}}$ -complete problem; by using the  $\text{EXP}^{\text{NP}}$ -completeness, one can translate the problem of computing  $\tilde{V}_\Phi(x)$  into the original problem in polynomial time, and hence we can ask the oracles to output  $\tilde{V}_\Phi(x)$ . Let  $f_0, f_1: \mathbb{F}^n \rightarrow \mathbb{F}$  denote the answers of the oracles  $A_0, A_1$ , respectively. Then, we have  $f_i = \tilde{V}_\Phi$  for an honest oracle  $A_i$ .

Although  $f_i$  is not necessarily multilinear for a dishonest oracle  $A_i$ , we can ensure that it is close to some multilinear function. This can be done by the multilinearity test, which was one of the main technical ingredients in the proof of  $\text{MIP} = \text{NEXP}$  [6]. For two functions  $f, g: \mathbb{F}^n \rightarrow \mathbb{F}$  and a real number  $\delta \in \mathbb{R}$ , we say that  $f$  and  $g$  are  $\delta$ -close if  $\Pr_{x \in \mathbb{F}^n} [f(x) \neq g(x)] < \delta$ .

► **Lemma 4.6** (Multilinearity Test [6]). *Let  $n \in \mathbb{N}$  and  $\mathbb{F}$  be a finite field. There exist a constant  $\delta = n^{O(1)}/|\mathbb{F}|$  and an efficient probabilistic algorithm that, given oracle access to an arbitrary function  $f: \mathbb{F}^n \rightarrow \mathbb{F}$ ,*

1. *accepts with probability 1 if  $f$  is multilinear, and*
2. *rejects with high probability if  $f$  is not  $\delta$ -close to any multilinear function.*

We perform the multilinearity test for  $f_0$  and  $f_1$ . Suppose that  $f_i$  is not  $\delta$ -close to any multilinear function for a dishonest oracle  $A_i$ . Then, the multilinearity test fails and hence we can doubt  $A_i$  with high probability. Therefore, in what follows, we may assume that both  $f_0$  and  $f_1$  are  $\delta$ -close to some multilinear functions  $\hat{f}_0$  and  $\hat{f}_1$ , respectively (note that  $\hat{f}_0$  and  $\hat{f}_1$  are unique for small  $\delta$ ).

In reality, we have only access to  $f_0, f_1$  instead of multilinear functions  $\hat{f}_0, \hat{f}_1$ . However, we may pretend to have access to the multilinear functions  $\hat{f}_0, \hat{f}_1$ , by using the random self-reducibility of multivariate low-degree polynomials (also known as the self-correction of the Reed-Muller code).

► **Lemma 4.7** (Self-correction; Beaver and Feigenbaum [8] and Lipton [21]). *There exists an efficient probabilistic algorithm that, given input  $x \in \mathbb{F}^n$  and oracle access to a function  $f: \mathbb{F}^n \rightarrow \mathbb{F}$  that is  $\delta$ -close to a multilinear function  $\hat{f}: \mathbb{F}^n \rightarrow \mathbb{F}$ , outputs  $\hat{f}(x)$  with probability at least  $1 - \delta(n+1)$ .*

**Proof.** Let  $a_0, \dots, a_n$  be arbitrary distinct points in  $\mathbb{F} \setminus \{0\}$ . Pick a random point  $y \in_R \mathbb{F}^n$ . By the polynomial interpolation, find the univariate polynomial  $p$  of degree at most  $n$  such that  $f(x + a_i \cdot y) = p(a_i)$  for all  $i \in \{0, \dots, n\}$ , and output  $p(0)$ .

Since  $x + a_i \cdot y$  is uniformly distributed on  $\mathbb{F}^n$  for any fixed  $x$  and  $a_i \neq 0$ , it holds that  $\hat{f}(x + a_i \cdot y) = f(x + a_i \cdot y)$  with probability at least  $1 - \delta$ . By the union bound, we have  $p(a_i) = \hat{f}(x + a_i \cdot y)$  for each  $i \in \{0, \dots, n\}$  with probability at least  $1 - \delta(n + 1)$ ; thus we have  $p(0) = \hat{f}(x)$  with probability at least  $1 - \delta(n + 1)$ , because  $\hat{f}$  is a polynomial of total degree at most  $n$ .  $\blacktriangleleft$

► **Remark 4.8.** In the case of the proof of  $\text{MIP} = \text{NEXP}$ , the self-correcting algorithm was not needed; for the sum-check protocol, it is sufficient to evaluate a multilinear function  $\hat{f}_i$  on random points  $x \in_R \mathbb{F}^n$ , rather than fixed points. In contrast, we need to evaluate a multilinear function  $\hat{f}_i$  on points that are not uniformly distributed, during the binary search.

In the following, we pretend that the dishonest oracle  $A_i$  asserts that the satisfying assignment is  $\hat{f}_i|_{\{0,1\}^n}$ , instead of  $f_i|_{\{0,1\}^n}$ . (Note that it holds that  $f_i|_{\{0,1\}^n} = \hat{f}_i|_{\{0,1\}^n} = V_\Phi$  for the honest oracle  $A_i$ .)

## Identifying the Larger Assignment

We are now ready to describe how to identify the larger assignment. It is sufficient to show that we can find, with high probability, the lexicographically first index  $z \in \{0, 1\}^n$  such that  $\hat{f}_0(z) \neq \hat{f}_1(z)$ .

First, we check if  $\hat{f}_0(b_{\text{in}}) = \hat{f}_1(b_{\text{in}})$ : For each  $i \in \{0, 1\}$ , run the self-correcting algorithm for  $f_i$  to obtain  $\hat{f}_i(b_{\text{in}})$ . If  $\hat{f}_0(b_{\text{in}}) = \hat{f}_1(b_{\text{in}})$ , then output it (which is surely the correct answer since  $\hat{f}_i(b_{\text{in}}) = V_\Phi(b_{\text{in}})$  for the honest oracle  $A_i$ ) and halt. Otherwise, perform the binary search described below.

We compute the lexicographically first disagreement  $z = (z_1, \dots, z_n) \in \{0, 1\}^n$  one by one. For  $j := 1$  to  $n$ , repeat the following: Suppose that we have computed  $z_1, \dots, z_{j-1}$ . Pick a random point  $u = (u_{j+1}, \dots, u_n) \in_R \mathbb{F}^{n-j}$  uniformly at random. Define  $x := (z_1, \dots, z_{j-1}, 0, u_{j+1}, \dots, u_n) \in \mathbb{F}^n$ . For each  $i \in \{0, 1\}$ , use the self-correcting algorithm for  $f_i$  to obtain  $\hat{f}_i(x)$ . If  $\hat{f}_0(x) \neq \hat{f}_1(x)$ , then set  $z_j := 0$ ; else, set  $z_j := 1$ .

► **Claim 4.9.** Assume that  $\hat{f}_0(b_{\text{in}}) \neq \hat{f}_1(b_{\text{in}})$ . Let  $z \in \{0, 1\}^n$  denote the lexicographically first index such that  $\hat{f}_0(z) \neq \hat{f}_1(z)$ . Then, the binary search described above correctly computes  $z$  with probability at least  $1 - \delta n(n + 1) - \frac{n^2}{|\mathbb{F}|}$ .

In particular, by setting  $|\mathbb{F}|$  large enough, we can compute  $z$  with high probability.

**Proof.** Let  $j \in \{1, \dots, n\}$ . Consider the  $j$ th iteration and assume that we have computed  $z_1, \dots, z_{j-1}$  correctly. For each  $i \in \{0, 1\}$ , let  $f'_i: \mathbb{F}^{n-j} \rightarrow \mathbb{F}$  be the multilinear function such that

$$f'_i(t_{j+1}, \dots, t_n) = \hat{f}_i(z_1, \dots, z_{j-1}, 0, t_{j+1}, \dots, t_n),$$

for any  $(t_{j+1}, \dots, t_n) \in \mathbb{F}^{n-j}$ . (The binary search tries to check if  $f'_0 \neq f'_1$  by the polynomial identity testing, and sets  $z_j := 0$  if and only if  $f'_0 \neq f'_1$ .)

If  $z_j = 0$ , then we have  $f'_0 \neq f'_1$  because  $f'_0(z_{j+1}, \dots, z_n) \neq f'_1(z_{j+1}, \dots, z_n)$ . The probability that the self-correcting algorithm outputs  $\hat{f}_i(x)$  correctly is at least  $1 - \delta(n + 1)$  for a dishonest oracle  $A_i$ . By the Schwartz-Zippel lemma, the probability that  $f'_0(u) \neq f'_1(u)$  for a random point  $u \in_R \mathbb{F}^{n-j}$  is at least  $1 - \frac{n-j}{|\mathbb{F}|} \geq 1 - \frac{n}{|\mathbb{F}|}$ . Therefore, the algorithm sets  $z_j := 0$  correctly with probability at least  $1 - \delta(n + 1) - \frac{n}{|\mathbb{F}|}$ .

If  $z_j = 1$ , then it follows from the minimality of  $z$  that  $f'_0(t) = f'_1(t)$  for every  $t \in \{0, 1\}^{n-j}$ . Since  $f'_0$  and  $f'_1$  are multilinear, we have  $f'_0 = f'_1$  by the uniqueness of the multilinear extension

(Proposition 4.5) and hence  $f'_0(u) = f'_1(u)$  holds for any  $u \in \mathbb{F}^{n-j}$ . Therefore, since the self-correcting algorithm outputs  $\hat{f}_i(x)$  with probability at least  $1 - \delta(n+1)$ , the algorithm sets  $z_j := 1$  correctly with probability at least  $1 - \delta(n+1)$ .

Overall, the algorithm computes  $z$  correctly with probability at least

$$\left(1 - \delta(n+1) - \frac{n}{|\mathbb{F}|}\right)^n \geq 1 - \delta n(n+1) - \frac{n^2}{|\mathbb{F}|}.$$

◀

We have computed the lexicographically first disagreement  $z \in \{0, 1\}$  such that  $\hat{f}_0(z) \neq \hat{f}_1(z)$ . Run the self-correcting algorithm to obtain  $\hat{f}_0(z)$  and  $\hat{f}_1(z)$ . Without loss of generality (by swapping the oracles if  $\hat{f}_0(z) > \hat{f}_1(z)$ ), we may assume that  $\hat{f}_0(z) < \hat{f}_1(z)$ .

Now we know, with high probability, that  $A_1$  asserts the larger (presumably satisfying) assignment  $\hat{f}_1|_{\{0,1\}^n} : \{0, 1\}^n \rightarrow \mathbb{F}$ .

### Verifying the Satisfiability

All that remains is to verify that  $\hat{f}_1|_{\{0,1\}^n}$  satisfies  $F_\Phi$ , which can be done in the same way with a proof of  $\text{MIP} = \text{NEXP}$ . For completeness, we sketch a proof suggested in [6, Section 7.1] and observe that it can be done with the help of an  $\text{EXP}^{\text{NP}}$ -complete oracle.

Babai, Fortnow, Lund [6] used the sum-check protocol [22] to check whether or not an exponentially long assignment satisfies  $F_\Phi$ . Basically, checking if an assignment  $\hat{f}_1|_{\{0,1\}^n} : \{0, 1\}^n \rightarrow \mathbb{F}$  satisfies a Boolean formula  $F_\Phi$  reduces to checking if some low-degree polynomials  $g : \mathbb{F}^l \rightarrow \mathbb{F}$  evaluate to 0 on  $\{0, 1\}^l$ .

Let us arithmetize the Boolean formula  $\Phi : \{0, 1\}^{m+3n+3} \rightarrow \{0, 1\}$  to a low-degree polynomial  $\tilde{\Phi} : \mathbb{F}^{m+3n+3} \rightarrow \mathbb{F}$  in the standard way, so that  $\Phi$  and  $\tilde{\Phi}$  agree on  $\{0, 1\}^{m+3n+3}$  (see [6, Section 3.1]). Define  $g^1 : \mathbb{F}^{m+3n} \rightarrow \mathbb{F}$  and  $g^2 : \mathbb{F}^n \rightarrow \mathbb{F}$  as

$$g^1(w) := 1 - \tilde{\Phi}(w, \hat{f}_1(b_1), \hat{f}_1(b_2), \hat{f}_1(b_3)), \quad (3)$$

$$g^2(b) := \hat{f}_1(b)(1 - \hat{f}_1(b)), \quad (4)$$

where  $w = (y, (b_1, b_2, b_3)) \in \mathbb{F}^m \times (\mathbb{F}^n)^3$  and  $b \in \mathbb{F}^n$ . Note that since  $\hat{f}_1$  and  $\tilde{\Phi}$  are low-degree polynomials, so are  $g^1$  and  $g^2$ .

It is easy to see that  $g^1(w) = 0$  and  $g^2(b) = 0$  for any  $w \in \{0, 1\}^{m+3n}$  and  $b \in \{0, 1\}^n$  and only if  $\hat{f}_1|_{\{0,1\}^n}$  is a satisfying assignment of  $F_\Phi$ . Indeed,  $g^2(b) = 0$  forces  $\hat{f}_1|_{\{0,1\}^n}$  to be a Boolean function (*i.e.*  $\hat{f}_1(b) \in \{0, 1\}$  for any  $b \in \{0, 1\}^n$ ), and  $g^1(w) = 0$  means that  $\Phi(w, \hat{f}_1(b_1), \hat{f}_1(b_2), \hat{f}_1(b_3))$  is true for any  $w \in \{0, 1\}^{m+3n}$ .

We note that, given a random point  $w$  or  $b$ , we can compute the value of  $g^1(w)$  or  $g^2(b)$  with high probability by substituting  $f_1$  for  $\hat{f}_1$  in (3) or (4) (*i.e.* we do not need to use the self-correcting algorithm); for a random point  $w \in_R \mathbb{F}^{m+3n}$ , it holds that  $g^1(w)$  computed by substituting  $f_1$  in (3) and  $g^1(w)$  are identical with probability at least  $1 - 3\delta$ .

Therefore, it is sufficient to show that we can check if each  $g \in \{g^1, g^2\}$  vanishes on  $\{0, 1\}^l$ , given access to a low-degree polynomial  $g$ . (Here,  $l := m + 3n$  if  $g = g^1$  and  $l := n$  if  $g = g^2$ .) There are several ways to verify that  $g : \mathbb{F}^l \rightarrow \mathbb{F}$  vanishes on  $\{0, 1\}^l$ , including [6, Section 7.1] and [5, 15, 9]. Here, we follow the way of Feige, Goldwasser, Lovász, Safra, and Szegedy [15].

We reduce a task of checking if  $g : \mathbb{F}^l \rightarrow \mathbb{F}$  vanishes on  $\{0, 1\}^l$  to a task of checking if a sum is equal to 0, the latter of which can be verified by the sum-check protocol (see [15,

Section 4.2.2] for more details): Pick a random point  $t = (t_1, \dots, t_l) \in_R \mathbb{F}^l$ . Consider the following sum:

$$\sum_{w=(w_1, \dots, w_l) \in \{0,1\}^l} g(w) \prod_{\{i|w_i=1\}} t_i = \sum_{w \in \{0,1\}^l} g(w) \prod_{i \in \{1, \dots, l\}} (w_i t_i + 1 - w_i). \tag{5}$$

If  $g$  vanishes on  $\{0, 1\}^l$ , then this sum is equal to 0. Otherwise, regarding the left-hand side of (5) as a multilinear function on variables  $t_1, \dots, t_l$ , the sum is not equal to 0 with probability at least  $1 - \frac{1}{|\mathbb{F}|}$  by the Schwartz-Zippel lemma. Therefore, by defining a low-degree polynomial  $h_t: \mathbb{F}^l \rightarrow \mathbb{F}$  as  $h_t(w) := g(w) \prod_{i \in \{1, \dots, l\}} (w_i t_i + 1 - w_i)$  for any  $w \in \mathbb{F}^l$ , it is sufficient to check if the sum of  $h_t(w)$  over  $w \in \{0, 1\}^l$  is equal to 0, which can be done by the sum-check protocol.

We describe the sum-check protocol briefly (see [6, Section 3.2] for a detailed description): In order to check if  $\sum_{w \in \{0,1\}^l} h_t(w) = 0$ , pick a random point  $r = (r_1, \dots, r_l) \in_R \mathbb{F}^l$ . Define a low-degree univariate polynomial  $g_i: \mathbb{F} \rightarrow \mathbb{F}$  for each  $i \in \{1, \dots, l\}$  as

$$g_i(x) := \sum_{(w_{i+1}, \dots, w_l) \in \{0,1\}^{l-i}} h_t(r_1, \dots, r_{i-1}, x, w_{i+1}, \dots, w_l)$$

and  $g_0(x) := 0$ . We request the oracle  $A_1$  to reveal all the coefficients of the univariate polynomial  $g_i$  for all  $i \in \{1, \dots, l\}$ . We trust  $A_1$  if and only if  $g_{i-1}(r_{i-1}) = g_i(0) + g_i(1)$  for each  $i \in \{1, \dots, l\}$  (*the Consistency Test*) and  $g_l(r_l) = h_t(r)$  (*the Final Test*). Here, since  $r$  is a random point, we may evaluate  $h_t(r)$  by using  $f_1$  in place of  $\hat{f}_1$  in (3) and (4).

We claim that the complexity of the honest oracle to output  $g_i$  is bounded by  $\text{EXP}^{\text{NP}}$ . Consider the following search problem: given a Boolean formula  $\Phi$ , a prime  $|\mathbb{F}|$ , and  $r, t \in \mathbb{F}^l$ , the task is to output all the coefficients of  $g_i$  for all  $i \in \{1, \dots, l\}$  (which can be written in a binary representation of polynomial length), where  $\widetilde{V}_\Phi$  is substituted for  $\hat{f}_1$  in (3) and (4). Regarding this problem as a decision problem, one can easily show that the problem is computable in  $\text{EXP}^{\text{NP}}$ . Thus, we can request the oracle  $A_1$  to output  $g_i$ .

Finally, we conclude the proof by analyzing the correctness (assuming that the binary search succeeded):

1. If  $A_1$  is honest, then  $\hat{f}_1 = f_1 = \widetilde{V}_\Phi$ . Thus, each  $g \in \{g^1, g^2\}$  vanishes on  $\{0, 1\}^l$ , and hence the sum (5) is 0; therefore, we can trust  $A_1$  with probability 1.
2. If  $A_1$  is dishonest, then  $\hat{f}_1$  does not constitute a satisfying assignment of  $F_\Phi$ . (If it were a satisfying assignment, then  $\hat{f}_1|_{\{0,1\}^n}$  would be a satisfying assignment larger than  $\hat{f}_0|_{\{0,1\}^n} = V_\Phi$ .) Thus, for some  $g \in \{g^1, g^2\}$ , the sum (5) is not 0 with probability at least  $1 - \frac{1}{|\mathbb{F}|}$ .

Assume that the sum is not 0, and let  $d \in \mathbb{N}$  be an upper bound on the degree of the low-degree polynomial  $h_t$ . Suppose that the dishonest oracle claimed that  $g_i$  is  $g'_i$  for each  $i \in \{1, \dots, l\}$ . Assuming that the Consistency Tests pass (*i.e.*  $g'_{i-1}(r_{i-1}) = g'_i(0) + g'_i(1)$  for each  $i \in \{1, \dots, l\}$ ), it holds that  $g'_i(r_l) \neq g_l(r_l) = h_t(r)$  with probability at least  $1 - \frac{dl}{|\mathbb{F}|}$  (see [6, Section 3.2]). The probability that  $h_t$  can be evaluated correctly on a random point  $r \in_R \mathbb{F}^l$  is at least  $1 - 3\delta$ . Thus, the Final Test (*i.e.*  $g'_l(r_l) = h_t(r)$ ) fails with probability at least  $1 - \frac{dl}{|\mathbb{F}|} - 3\delta$ .

Overall, we can doubt  $A_1$  with probability at least  $1 - \frac{dl}{|\mathbb{F}|} - 3\delta - \frac{1}{|\mathbb{F}|}$ .

## 5 Deterministic Selector

This section is devoted to investigating a deterministic selector.

To prove the existence of a deterministic selector for a PSPACE-complete language (Theorem 1.4, Part 1), we show that a deterministic selector can be constructed based on downward self-reducibility:

► **Theorem 5.1.** *Any downward self-reducible language has a deterministic selector.*

Since there exists a downward self-reducible PSPACE-complete language, we immediately obtain a deterministic selector for any PSPACE-complete language.

**Proof.** Let  $L$  be a downward self-reducible language. Namely, there exists a polynomial-time oracle machine  $M$  such that

- $M^L(x) = L(x)$  for any  $x \in \{0, 1\}^*$ , and
- $M$  does not make any queries of length greater than or equal to  $|x|$ , on input  $x \in \{0, 1\}^*$ .

The idea is to keep a string  $y$  such that  $A_0(y) \neq A_1(y)$ , and to run  $M^{A_0}$  and  $M^{A_1}$  to obtain another string  $q$  of length less than  $|y|$  such that  $A_0(q) \neq A_1(q)$ . Consider the following algorithm: Given an input  $x \in \{0, 1\}^*$  and two oracles  $A_0, A_1$ , if  $A_0(x) = A_1(x)$  then output it and halt. Else, let  $y := x$  and repeat the following: Compute  $M^{A_i}(y)$  for each  $i \in \{0, 1\}$ . If  $M^{A_0}(y) = M^{A_1}(y) =: b$ , then we trust the oracle  $A_i$  such that  $A_i(y) = b$  and output  $A_i(x)$ . Otherwise, let  $q$  be the first query that  $M^{A_0}$  and  $M^{A_1}$  make on input  $y$  such that  $A_0(q) \neq A_1(q)$ . (There exists such a  $q$  because  $M^{A_0}(y) \neq M^{A_1}(y)$ ; moreover, it holds that  $|q| < |y|$  by the definition of downward self-reducibility.) Then, we update  $y := q$  and move on to the next iteration.

This algorithm runs in polynomial time, since  $|y|$  decreases in each repetition.

We claim the correctness of the algorithm. It is easy to see that  $A_0(y) \neq A_1(y)$  at the beginning of each repetition. Suppose that  $M^{A_0}(y) = M^{A_1}(y) =: b$ . Since  $A_0$  or  $A_1$  is equal to  $L$ , we have  $b = M^{A_0}(y) = M^{A_1}(y) = M^L(y) = L(y)$ , where the last equality holds by the definition of  $M$ . Moreover, there exists the unique  $i \in \{0, 1\}$  such that  $A_i(y) = b$  because  $A_0$  and  $A_1$  disagree on  $y$ . Therefore,  $A_i$  is honest if and only if  $A_i(y) = b (= L(y))$ . ◀

Then, we claim that any language with a deterministic selector is in PSPACE (Theorem 1.4, Part 2). We thereby prove that the supremum of the languages with a deterministic selector is PSPACE.

**Proof of Part 2 of Theorem 1.4.** Let  $L$  be a language with a deterministic selector  $S$ .

The idea is to regard a computation of  $S$  as a game played between the NO player and the YES player (which correspond to two oracles  $A_0$  and  $A_1$ , respectively): On input  $x \in \{0, 1\}^*$ , the YES player tries to convince the selector  $S$  that  $x \in L$ , whereas the NO player tries to convince  $S$  that  $x \notin L$ . The YES player chooses  $A_1 \subseteq \{0, 1\}^*$  such that  $x \in A_1$ , and the NO player chooses  $A_0 \subseteq \{0, 1\}^*$  such that  $x \notin A_0$ . Then, we simulate  $S^{A_0, A_1}(x)$ , and the YES player wins if and only if  $S^{A_0, A_1}(x) = 1$ .

It is easy to see that the YES player has a winning strategy if  $x \in L$ . Indeed, the YES player wins by setting  $A_1 = L$ ; similarly, if  $x \notin L$ , then the NO player wins by setting  $A_0 = L$ . Therefore, it is sufficient to show that we can compute the player that has a winning strategy in PSPACE.

We may restate the game as follows: Simulate  $S$  on input  $x$ . If  $S$  makes a query  $x$  to  $A_i$  ( $i \in \{0, 1\}$ ), then answer it with  $i$ . If  $S$  makes a query  $q$  ( $\neq x$ ) to the oracle  $A_0$ , then the NO player gives an arbitrary answer; similarly, if  $S$  makes a query to  $A_1$ , then the YES player gives an arbitrary answer. (However, we require the players to behave in a consistent way: if  $S$  makes the same query more than once, then a player must give the same answer that the player answered in the past.)



Again, one can easily prove that the YES player has a winning strategy for this game if and only if  $x \in L$ .

Now we describe a polynomial-time alternating Turing machine that computes  $L$ : Simulate the game described above, while universally guessing the answers of the NO player and existentially guessing the answers of the YES player. Since a polynomial-time alternating machine can be simulated in PSPACE, it holds that  $L \in \text{PSPACE}$ . ◀

## 6 Random Strings vs. Randomized Computation

In this section, we apply the notion of selector to the proof by Buhrman, Fortnow, Koucký, and Loff [11]. We thereby extend their result from  $\{\text{NP}, \text{P}^{\#P}, \text{PSPACE}, \text{EXP}\}$  to any classes whose complete languages have a selector (e.g.  $\Sigma_i^P, \Pi_i^P, \text{P}^{\#P}, \text{PSPACE}, \text{EXP}$ , and  $\text{EXP}^{\text{NP}}$ ).

► **Theorem 6.1** (Extended Theorem 15 of [11]). *Let  $\alpha: \{0\}^* \rightarrow \{0, 1\}^*$  be a length preserving function,  $c > 0$  be a constant such that  $\alpha(0^n) \notin \text{i.o-EXP}/n - c \log n$ , and  $\mathcal{C}$  be a complexity class such that there is a selector for some paddable  $\mathcal{C}$ -complete language  $L$ . If  $L \in \text{P}/\alpha(0^{n^d})$  for some  $d > 0$ , then  $\mathcal{C} \subseteq \text{BPP}$ .*

**Proof.** Let  $M$  be a polynomial-time machine such that  $L(x) = M(x, \alpha(0^{|x|^d}))$ , and  $G_n \subseteq \{0, 1\}^{n^d}$  be the set of “good” advice:

$$G_n := \{r \in \{0, 1\}^{n^d} \mid \forall x \in \{0, 1\}^n, L(x) = M(x, r)\}.$$

Buhrman *et al.* [11] showed that  $|G_n| \geq 2^{n^d}/n^{cd}$  by exploiting the high nonuniform complexity of advice  $\alpha(0^{n^d})$ .

As with Theorem 2.8, there exist a polynomial  $l$  and a selector  $S$  that identifies an honest oracle among  $m := 2l(n)^{cd}$  oracles with probability at least  $\frac{5}{6}$ , and makes only queries of length exactly  $l(n)$  on inputs of length  $n$ .

Consider the following probabilistic algorithm: On input  $x \in \{0, 1\}^n$ , let  $l$  denote  $l(n)$ . We pick  $m$  random strings  $r_1, \dots, r_m \in_R \{0, 1\}^{l^d}$  uniformly at random, and define oracles  $A_i(q) = M(q, r_i)$ , for any  $i \in \{1, \dots, m\}$  and for any  $q \in \{0, 1\}^l$ . We simulate  $S$  on input  $x$ , answering its queries  $q \in \{0, 1\}^l$  to  $A_i$  by computing  $M(q, r_i)$ .

The probability that we fail to pick any “good” advice, namely  $r_i \notin G_l$  for all  $i$ , is  $(1 - |G_l|)^{2l^{cd}} \leq e^{-2l^{cd}/l^{cd}} < \frac{1}{6}$ . Thus, we can output the correct answer with probability at least  $\frac{2}{3}$  overall. ◀

## 7 Concluding Remarks

We state some open problems and possible directions for future work:

- Do there exist selectors for NEXP-complete languages or promise- $S_2^{\text{EXP}}$ -complete languages? In particular, it is interesting to close the gap between  $\text{EXP}^{\text{NP}}$  and  $S_2^{\text{EXP}}$ : although these classes seem “close” in some sense,  $\text{EXP}^{\text{NP}}$  and  $S_2^{\text{EXP}}$  are very different in the known relationship with BPP; it is a notorious open problem whether  $\text{BPP} \neq \text{EXP}^{\text{NP}}$ , whereas one can prove  $\text{BPP} \neq S_2^{\text{EXP}}$ .
- We proved that a property of removing short advice can be captured by the notion of selector. What about a property of removing advice of polynomial length?
- The result of  $\text{MIP} = \text{NEXP}$  was “scaled-down” to obtain the relationship with hardness of approximating cliques [15], and eventually the PCP theorem [4, 3] was established. Can we obtain such interesting applications of selectors, by scaling down the selector for  $\text{EXP}^{\text{NP}}$ -complete languages?

**Acknowledgements.** I greatly appreciate Hiroshi Imai’s advice and comments that significantly improved the presentation; I thank Akitoshi Kawamura for many useful discussions; I am deeply grateful to Lance Fortnow and the anonymous CCC reviewers for very helpful comments that made the paper more understandable; and I would like to thank the reviewer for suggesting the title.

---

## References

- 1 Scott Aaronson and Avi Wigderson. Algebrization: A new barrier in complexity theory. *ACM Trans. Comput. Theory*, 1(1):2:1–2:54, 2009.
- 2 Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 1st edition, 2009.
- 3 Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *J. ACM*, 45(3):501–555, 1998.
- 4 Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: A new characterization of NP. *J. ACM*, 45(1):70–122, 1998.
- 5 László Babai, Lance Fortnow, Leonid Levin, and Mario Szegedy. Checking computations in polylogarithmic time. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing*, STOC’91, pages 21–32, 1991.
- 6 László Babai, Lance Fortnow, and Carsten Lund. Non-deterministic exponential time has two-prover interactive protocols. *Comput. Complex.*, 1:3–40, 1991.
- 7 Boaz Barak. A probabilistic-time hierarchy theorem for “slightly non-uniform” algorithms. In *Proceedings of the 6th International Workshop on Randomization and Approximation Techniques*, RANDOM’02, pages 194–208, 2002.
- 8 Donald Beaver and Joan Feigenbaum. Hiding instances in multioracle queries. In *Proceedings of the 7th Annual Symposium on Theoretical Aspects of Computer Science*, STACS’90, pages 37–48, 1990.
- 9 Eli Ben-Sasson and Madhu Sudan. Short PCPs with polylog query complexity. *SIAM J. Comput.*, 38(2):551–607, 2008.
- 10 Manuel Blum and Sampath Kannan. Designing programs that check their work. *J. ACM*, 42(1):269–291, 1995.
- 11 Harry Buhrman, Lance Fortnow, Michal Koucký, and Bruno Loff. Derandomizing from random strings. In *Proceedings of the 25th Annual Conference on Computational Complexity*, CCC’10, pages 58–63, 2010.
- 12 Harry Buhrman, Lance Fortnow, and Rahul Santhanam. Unconditional lower bounds against advice. In *Proceedings of the 36th International Colloquium on Automata, Languages, and Programming*, ICALP’09, pages 195–209, 2009.
- 13 Harry Buhrman and Steven Homer. Superpolynomial circuits, almost sparse oracles and the exponential hierarchy. In *Proceedings of the 12th Conference on Foundations of Software Technology and Theoretical Computer Science*, FSTTCS’92, pages 116–127, 1992.
- 14 Jin-Yi Cai.  $S_2^p \subseteq \text{ZPP}^{\text{NP}}$ . *J. Comput. Syst. Sci.*, 73(1):25–35, 2007.
- 15 Uriel Feige, Shafi Goldwasser, Laszlo Lovász, Shmuel Safra, and Mario Szegedy. Interactive proofs and the hardness of approximating cliques. *J. ACM*, 43(2):268–292, 1996.
- 16 Lance Fortnow and Adam Klivans. NP with small advice. In *Proceedings of the 20th Annual Conference on Computational Complexity*, CCC’05, pages 228–234, 2005.
- 17 Lance Fortnow, John Rempel, and Michael Sipser. On the power of multi-prover interactive protocols. *Theor. Comput. Sci.*, 134(2):545–557, 1994.
- 18 Richard Karp and Richard Lipton. Turing machines that take advice. *Enseign. Math*, 28(2):191–209, 1982.



- 19 Adam Klivans and Dieter van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses. *SIAM J. Comput.*, 31(5):1501–1526, 2002.
- 20 Mark Krentel. The complexity of optimization problems. *J. Comput. Syst. Sci.*, 36(3):490–509, 1988.
- 21 Richard Lipton. New directions in testing. In Joan Feigenbaum and Michael Merritt, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, volume 2, pages 191–202. American Mathematical Society, 1991.
- 22 Carsten Lund, Lance Fortnow, Howard Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. *J. ACM*, 39(4):859–868, 1992.
- 23 Alexander Russell and Ravi Sundaram. Symmetric alternation captures BPP. *Comput. Complex.*, 7(2):152–162, 1998.
- 24 Adi Shamir.  $IP = PSPACE$ . *J. ACM*, 39(4):869–877, 1992.
- 25 Luca Trevisan and Salil Vadhan. Pseudorandomness and average-case complexity via uniform reductions. *Comput. Complex.*, 16(4):331–364, 2007.

# Adaptivity Helps for Testing Juntas

Rocco A. Servedio<sup>1</sup>, Li-Yang Tan<sup>2</sup>, and John Wright<sup>3</sup>

- 1 Columbia University  
New York, USA  
rocco@cs.columbia.edu
- 2 Simons Institute, UC Berkeley  
Berkeley, USA  
liyang@cs.columbia.edu
- 3 Carnegie Mellon University  
Pittsburgh, USA  
jswright@cs.cmu.edu

---

## Abstract

We give a new lower bound on the query complexity of any non-adaptive algorithm for testing whether an unknown Boolean function is a  $k$ -junta versus  $\varepsilon$ -far from every  $k$ -junta. Our lower bound is that any non-adaptive algorithm must make

$$\Omega\left(\frac{k \log k}{\varepsilon^c \log(\log(k)/\varepsilon^c)}\right)$$

queries for this testing problem, where  $c$  is any absolute constant  $< 1$ . For suitable values of  $\varepsilon$  this is asymptotically larger than the  $O(k \log k + k/\varepsilon)$  query complexity of the best known adaptive algorithm [9] for testing juntas, and thus the new lower bound shows that adaptive algorithms are more powerful than non-adaptive algorithms for the junta testing problem.

**1998 ACM Subject Classification** F.2 Analysis of Algorithms and Problem Complexity

**Keywords and phrases** Property testing, juntas, adaptivity

**Digital Object Identifier** 10.4230/LIPIcs.CCC.2015.264

## 1 Introduction

As popular and scientific interest in “big data” continues to build, the field of sublinear-time algorithms has received increasing research attention in recent years. The study of *property testing* is an important area within sublinear algorithms. At a high level, property testing algorithms are “ultra-fast” randomized algorithms which aim to (approximately) determine whether an unknown “massive object” has a particular property while inspecting only a tiny (sublinear, or in some cases even constant sized) portion of the object. Testing algorithms have by now been studied for many different types of mathematical objects; see e.g. [38, 39, 28] for some fairly recent surveys and overviews of contemporary property testing research.

In this work we shall consider property testing algorithms for Boolean functions, and in particular we study the question of testing whether an unknown Boolean function is a  $k$ -junta. Recall that a function  $f$  is a  $k$ -junta if it has at most  $k$  relevant variables, i.e. there exist  $k$  distinct indices  $i_1, \dots, i_k$  and a  $k$ -variable function  $g : \{0, 1\}^k \rightarrow \{0, 1\}$  such that  $f(x) = g(x_{i_1}, \dots, x_{i_k})$  for all  $x \in \{0, 1\}^n$ . A testing algorithm for  $k$ -juntas is given as input  $k$  and  $\varepsilon > 0$ , and is provided with black-box oracle access to an unknown and arbitrary  $f : \{0, 1\}^n \rightarrow \{0, 1\}$ . The algorithm must output “yes” with high probability (say at least  $2/3$ ) if  $f$  is a  $k$ -junta, and must output “no” with high probability if  $f$  disagrees with every



© Rocco A. Servedio, Li-Yang Tan, and John Wright;  
licensed under Creative Commons License CC-BY  
30th Conference on Computational Complexity (CCC'15).  
Editor: David Zuckerman; pp. 264–279



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

$k$ -junta on at least an  $\varepsilon$  fraction of all possible inputs. The main goal in property testing is to obtain algorithms which make as few queries as possible to the unknown black-box function.

We motivate our work by observing that juntas are a very basic type of Boolean function whose study intersects many different areas within theoretical computer science. In complexity theory and cryptography,  $k = O(1)$ -juntas are precisely the Boolean functions computed by  $\text{NC}^0$  circuits. Juntas arise naturally in settings where a small (unknown) set of features determines the label of a high-dimensional data point, and hence many researchers in learning theory have studied juntas across a wide range of different learning models, see e.g. [15, 22, 16, 30, 3, 37, 4, 2, 25, 42, 21]. Finally, the problem of testing whether an unknown Boolean function is a  $k$ -junta is one of the most thoroughly studied questions in Boolean function property testing. We briefly survey relevant previous work on testing juntas in the following subsection.

### 1.1 Prior work on testing juntas

Fischer et al. [26] were the first to explicitly consider the junta testing problem. Their influential paper gave several algorithms for testing  $k$ -juntas, the most efficient of which is a non-adaptive tester that makes  $O(k^2(\log k)^2/\varepsilon)$  queries. This was improved by Blais [8] who gave a non-adaptive testing algorithm that uses only  $O(k^{3/2}(\log k)^3/\varepsilon)$  queries; this result is still the most efficient known non-adaptive junta tester. Soon thereafter Blais [9] gave an *adaptive* junta testing algorithm that uses only  $O(k \log k + k/\varepsilon)$  queries, which remains the most efficient known junta testing algorithm to date.

We note that ideas and techniques from these junta testing algorithms have played an important role in a broad range of algorithmic results for other Boolean function property testing problems. These include efficient algorithms for testing various classes of functions, such as  $s$ -term DNF formulas, small Boolean circuits, and sparse  $GF(2)$  polynomials, that are close to juntas but not actually juntas themselves (see e.g. [23, 29, 24, 19]), as well as algorithms for testing linear threshold functions [34] (which in general are not close to juntas). Junta testing is also closely related to the problem of Boolean function isomorphism testing, see e.g. [13, 14, 18, 1].

Lower bounds for testing  $k$ -juntas have also been intensively studied. The original [26] paper gave an  $\Omega(\sqrt{k}/\log k)$  lower bound for nonadaptive algorithms that test whether an unknown function is a  $k$ -junta versus constant-far from every  $k$ -junta. Chockler and Gutfreund [20] simplified, strengthened and extended this lower bound by proving 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. (We describe the construction and sketch the [20] argument in Section 1.3 below). Blais [8] was the first to give a lower bound that involves the distance parameter  $\varepsilon$ ; he showed that for  $\varepsilon \geq k/2^k$ , any non-adaptive algorithm for  $\varepsilon$ -testing  $k$ -juntas must make  $\Omega\left(\frac{k}{\varepsilon \log(k/\varepsilon)}\right)$  queries.

In recent years numerous other works have given junta testing lower bounds. In [11] Blais, Brody and Matulef established a connection between lower bounds in communication complexity and property testing lower bounds, and used this connection (together with known lower bounds on the communication complexity of the size- $k$  set disjointness problem) to give a different proof of an  $\Omega(k)$  lower bound for adaptively testing whether a function is a  $k$ -junta versus constant-far from every  $k$ -junta. More recently, Blais, Brody and Ghazi [10] gave new bounds on the communication complexity of the Hamming distance function, and used these bounds to give an alternate proof of the  $\Omega(k)$  lower bound for adaptive junta testing algorithms via the [11] connection. Blais and Kane [12] studied the problem of

testing whether an  $n$ -variable Boolean function is a size- $k$  parity function (as noted in [12], lower bounds for this problem give lower bounds for testing juntas), and via a geometric and Fourier-based analysis gave a  $k - o(k)$  lower bound for adaptive algorithms and a  $2k - O(1)$  lower bound for non-adaptive algorithms, again for  $\varepsilon$  constant. Buhrman et al. [17] combined the communication complexity based approach of [11] with an  $\Omega(k \log k)$  lower bound for the one-way communication complexity of  $k$ -disjointness to obtain an  $\Omega(k \log k)$  lower bound (for constant  $\varepsilon$ ) for testing whether a function  $f$  is a size- $k$  parity, and hence for testing whether  $f$  is a  $k$ -junta.

## 1.2 Our main result: Adaptivity helps for testing juntas

While the junta testing problem has been intensively studied, the results described above still leave a gap between the query complexity of the best *adaptive* algorithm [9] and the strongest known lower bounds for *non-adaptive* junta testing. The lower bounds of  $\Omega\left(\frac{k}{\varepsilon \log(k/\varepsilon)}\right)$  from [8] and  $\Omega(k \log k)$  (for  $\varepsilon$  constant) from [17] are incomparable, but neither of them is strong enough, for any setting of  $\varepsilon$ , to exceed the  $O(k \log k + k/\varepsilon)$  upper bound from [9]. In [8] Blais asked as an open question “*Is there a gap between the query complexity of adaptive and non-adaptive algorithms for testing juntas?*” This question was reiterated in a 2010 survey article on testing juntas, in which Blais explicitly asked “*Does adaptivity help when testing  $k$ -juntas?*”, referring to this as a “basic problem” [7].

Our main contribution in the present work is to give a better lower bound on non-adaptive junta testing algorithms which implies that the answer to the above questions is “yes.” We prove the following:

► **Theorem 1.1.** *Let  $\mathcal{A}$  be any non-adaptive algorithm which tests whether an unknown black-box  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  is a  $k$ -junta versus  $\varepsilon$ -far from every  $k$ -junta. Then for all  $\varepsilon$  satisfying  $k^{-o_k(1)} \leq \varepsilon \leq o_k(1)$ , algorithm  $\mathcal{A}$  must make at least*

$$q = \frac{Ck \log k}{\varepsilon^c \log(\log(k)/\varepsilon^c)} \tag{1}$$

*queries, where  $c$  is any absolute constant  $< 1$  and  $C > 0$  is an absolute constant.*

For suitable choices of  $\varepsilon$ , such as  $\varepsilon = 1/(\log k)$ , the lower bound of Theorem 1.1 is asymptotically larger than the  $O(k \log k + k/\varepsilon)$  upper bound of the [9] adaptive algorithm. Thus, together with the [9] upper bound, our lower bound gives an affirmative answer to the question posed in [8, 7]: adaptivity helps for testing  $k$ -juntas.<sup>1</sup>

It is interesting that while all of the recent junta testing lower bounds [11, 10, 17] employ the connection with communication complexity lower bounds that was established in [11], our proof of Theorem 1.1 does not follow this approach. Instead, we give a proof using Yao’s classic minimax principle; however, our argument is somewhat involved, employing a new Boolean isoperimetric inequality and a very delicate application of a variant of McDiarmid’s “method of bounded differences” that allows for a (low-probability) bad event. In the rest of this section we motivate and explain our approach at a high level before giving the full proof in the subsequent sections.

---

<sup>1</sup> We note in this context that several other natural Boolean function classes are known to exhibit a gap between the query complexity of adaptive versus non-adaptive testing algorithms. These include the class of signed majority functions [35, 40] and the class of read-once width-two OBDD [41]. In all three cases the adaptive tester which beats the best possible non-adaptive tester may be viewed as performing some sort of binary search.

### 1.3 The idea underlying our proof

Our approach is inspired by the lower bound of Chockler and Gutfreund [20] for adaptive algorithms, so we begin by briefly recalling their construction and analysis. Chockler and Gutfreund define two distributions  $\mathcal{D}_{yes}$  and  $\mathcal{D}_{no}$  over  $(k+1)$ -variable functions. A random  $\mathbf{f}_{yes} \sim \mathcal{D}_{yes}$  is drawn by first choosing a random coordinate  $i \in [k+1]$  to be the irrelevant variable, and then choosing a random  $k$ -junta over the other  $k$  variables from  $x_1, \dots, x_{k+1}$ . A random  $\mathbf{f}_{no} \sim \mathcal{D}_{no}$  is drawn by choosing a random  $(k+1)$ -junta. Clearly every  $f$  in the support of  $\mathcal{D}_{yes}$  is a  $k$ -junta, and it is easy to show (for  $k$  larger than an absolute constant) that almost every function in the support of  $\mathcal{D}_{no}$  is constant-far from every  $k$ -junta.

Chockler and Gutfreund argue that any  $k/6$ -query *adaptive* algorithm  $\mathcal{A}$  must have

$$\left| \Pr_{\mathbf{f}_{yes} \sim \mathcal{D}_{yes}} [\mathcal{A} \text{ accepts } \mathbf{f}_{yes}] - \Pr_{\mathbf{f}_{no} \sim \mathcal{D}_{no}} [\mathcal{A} \text{ accepts } \mathbf{f}_{no}] \right| \leq \frac{1}{6},$$

which gives their  $\Omega(k)$  lower bound for adaptive algorithms. Their analysis shows that the only way an algorithm can get statistical evidence that the black-box  $f$  is a yes-function rather than a no-function is by querying a pair of inputs  $x, y \in \{0, 1\}^{k+1}$  that differ in precisely the coordinate  $i \in [k+1]$  that was chosen to be irrelevant in the selection of  $\mathbf{f}_{yes} \sim \mathcal{D}_{yes}$  (they refer to such a pair of Hamming neighbors  $x, x^{\oplus i}$  in  $\{0, 1\}^{k+1}$  as an *i-twin*). While we do not repeat their analysis here, for intuition we observe that if  $x, y$  form a  $j$ -twin for  $j \neq i$  then for both a random yes-function and a random no-function  $f(x) = f(y)$  with probability exactly  $1/2$ , while if  $x, y$  form an  $i$ -twin then  $f(x) = f(y)$  for a random yes-function with probability 1 while  $f(x) = f(y)$  with probability  $1/2$  for a random no-function. Since a set of  $t$  queries can contain  $i$ -twins for at most  $t - 1$  distinct coordinates, the  $\Omega(k)$  lower bound follows by a “needle in a haystack” argument.

The starting point of our work is the simple observation that the analysis of the Chockler-Gutfreund construction is tight for adaptive algorithms: there is an adaptive algorithm that can distinguish a random  $\mathbf{f}_{yes} \sim \mathcal{D}_{yes}$  from a random  $\mathbf{f}_{no} \sim \mathcal{D}_{no}$  with  $O(k)$  queries. This algorithm works as follows: for each successive coordinate  $j = 1, \dots, k+1$ , it draws random  $j$ -twins until either (a) a  $j$ -twin  $x, x^{\oplus j}$  is drawn for which  $f(x) \neq f(x^{\oplus j})$ , or (b)  $10 \log(k+1)$   $j$ -twins have been drawn and all had  $f(x) = f(x^{\oplus j})$ . If (b) holds for any  $j \in [k+1]$  then halt and output “ $k$ -junta,” and if (a) holds for every  $j \in [k+1]$  halt and output “not a  $k$ -junta.” Since the expected number of  $j$ -twins drawn for a coordinate  $j \neq i$  is 2, a straightforward analysis establishes that this algorithm wvhp makes  $O(k)$  queries and outputs the correct answer.

Intuitively, the above-described algorithm is only able to achieve  $O(k)$  query complexity (an amortized  $O(1)$  queries for each of the  $k+1$  coordinates) because it is *adaptive* and hence can stop querying a given coordinate  $j$  once it receives a  $j$ -twin with  $f(x) \neq f(x^{\oplus j})$ . Since there are  $k+1$  coordinates to consider, it is very likely that for some coordinate  $j \neq i$ , a collection of  $\frac{1}{2} \log k$  randomly selected  $j$ -twins will all have  $f(x) = f(x^{\oplus j})$  (in fact we expect this to happen for  $\approx \sqrt{k}$  different coordinates). Since non-adaptive algorithms cannot “amortize” the coordinates along which they spend their queries, this suggests that (i) any *nonadaptive* algorithm will need to query  $\Omega(\log k)$   $j$ -twins for at least  $\Omega(k)$  many choices of  $j \in [k+1]$ , and further raises the possibility that (ii) any non-adaptive algorithm for distinguishing  $\mathcal{D}_{yes}$  from  $\mathcal{D}_{no}$  may need  $\Omega(k \log k)$  queries.

In fact, (i) above is correct but (ii) is not. While indeed a non-adaptive algorithm must “rule out” at least  $\Omega(k)$  coordinates as not being irrelevant, and indeed  $\Omega(\log k)$   $j$ -twins must be queried to rule out a given coordinate  $j$  with confidence  $1 - 1/\text{poly}(k)$ , it does not follow that  $\Omega(k \log k)$  queries are required to rule out all coordinates. This is because a set of  $q$

query points can induce  $\omega(q)$  different twins – or, to put it in the more combinatorial terms that we use henceforth in the paper, a subset  $Q$  of vertices of the Boolean hypercube can induce  $\omega(|Q|)$  hypercube edges.<sup>2</sup> Indeed, as observed by Frankl [27], there is a set  $Q$  of only  $\Theta(k \frac{\log k}{\log \log k})$  points in  $\{0, 1\}^{k+1}$  that induces at least  $\log(k+1)$  edges along each of the  $k+1$  coordinates. This set  $S$  is as follows: letting  $\ell = \log(2 \log(k+1))$  (and assuming that  $\ell$  and  $\frac{k+1}{\ell}$  are integers), we partition  $[k+1]$  into sets  $A_1, \dots, A_{(k+1)/\ell}$  of equal size  $\ell$  each, and let  $Q$  be the union of the  $\frac{k+1}{\ell}$  subcubes  $C_1, \dots, C_{(k+1)/\ell}$  where  $C_i$  consists of all  $2^\ell$  strings whose 1-coordinates are all contained in positions in  $A_i$ . It is easy to verify that the corresponding non-adaptive algorithm makes  $\Theta(k \frac{\log k}{\log \log k})$  queries and successfully distinguishes  $\mathbf{f}_{yes} \sim \mathcal{D}_{yes}$  from  $\mathbf{f}_{no} \sim \mathcal{D}_{no}$ .

It turns out that this is indeed an optimal query lower bound for non-adaptive algorithms that distinguish  $\mathcal{D}_{yes}$  from  $\mathcal{D}_{no}$ , up to constant factors; this follows as a special case of our main result, taking  $\varepsilon$  to be constant. Our main result is proved by analyzing an  $\varepsilon$ -biased generalization of the Chockler-Gutfreund yes- and no-distributions; the distributions we consider are the same ones that Blais uses in [8] to establish his lower bound for non-adaptive algorithms. The analysis of [8] uses the edge-isoperimetric inequality of Harper [31], Bernstein [6], Lindsey [33], and Hart [32] and leads to a lower bound of  $\Omega\left(\frac{k}{\varepsilon \log(k/\varepsilon)}\right)$  queries for non-adaptive algorithms. In contrast, we use a different edge-isoperimetric inequality, which may be viewed as an extension of Frankl’s Theorem 4 in [27] (see Section 2.2). Our edge-isoperimetric inequality, which we state and prove in Section 2.2, implies that any set of vertices in  $\{0, 1\}^{k+1}$  that induces  $\Omega(\log k)$  edges in each of  $\Omega(k)$  distinct coordinates must be of size  $\Theta(k \frac{\log k}{\log \log k})$ .

Another significant difference between our approach and that of [8] is that while [8] essentially applies the Harper–Bernstein–Lindsey–Hart isoperimetric inequality via a union bound in a fairly straightforward way to obtain the  $\Omega\left(\frac{k}{\varepsilon \log(k/\varepsilon)}\right)$  lower bound, our argument yielding a  $\Omega\left(\frac{k \log k}{\varepsilon^c \log(\log(k)/\varepsilon^c)}\right)$  lower bound is significantly more involved. (The union bound approach of [8] would cost us at least a  $\log k$  factor, which is more than we can afford to separate adaptive versus non-adaptive query complexity.) Instead, we use our edge-isoperimetric inequality in the context of a careful probabilistic analysis (to bound the variation distance between “yes-function” and “no-function” vectors of responses a la Yao’s minimax method) which crucially relies on a variant of McDiarmid’s “method of bounded differences” in which a low-probability “bad event” may take place [36].

## 1.4 Preliminaries

All logarithms are base 2 unless otherwise stated. We use boldface (e.g.  $\mathbf{x}$ ,  $\mathbf{y}$ , and  $\mathbf{f}$ ) to denote random variables. Given  $S \subseteq \{0, 1\}^n$ , we write  $G_S$  to denote the subgraph of the Hamming graph induced by  $S$ . That is,  $G_S = (S, E_S)$ , where  $(x, y) \in E_S$  iff  $x, y \in S$  and  $x = y^{\oplus i}$  (this is the string obtained by flipping  $y$  in the  $i$ -th coordinate) for some  $i \in [n]$ ; we call such an edge  $(x, y)$  an  $i$ -edge induced by  $S$ .

<sup>2</sup> The edge-isoperimetric inequality of Harper [31], Bernstein [6], Lindsey [33], and Hart [32] gives a tight upper bound of  $\frac{1}{2}|Q| \log |Q|$  edges. We return to this in Section 2.2 when we state and prove a different edge-isoperimetric inequality that we need for our proof.

**2 Proof of Theorem 1.1**

**2.1 The “yes” and “no” distributions**

As discussed in the introduction, we consider the same distributions  $\mathcal{D}_{yes}$  and  $\mathcal{D}_{no}$  that Blais used in [8] to establish his non-adaptive lower bound, which are biased generalizations of the yes- and no-distributions considered by Chockler and Gutfreund in [20]. A draw from  $\mathcal{D}_{no}$  is an “ $\varepsilon$ -biased random  $(k + 1)$ -junta”  $\mathbf{f}_{no} : \{0, 1\}^{k+1} \rightarrow \{0, 1\}$ , one which independently takes value 1 with probability  $\varepsilon$  on every string in  $\{0, 1\}^{k+1}$ . A random  $\mathbf{f}_{yes}$  from  $\mathcal{D}_{yes}$  is drawn by first choosing a random coordinate  $i \in [k + 1]$  to be irrelevant, and then choosing a random  $\varepsilon$ -biased random  $k$ -junta over the variables from  $\{x_1, \dots, x_{k+1}\} \setminus \{x_i\}$ . Equivalently,  $\mathcal{D}_{yes}$  is the uniform mixture of  $\mathcal{D}_{yes}^{(1)}, \dots, \mathcal{D}_{yes}^{(k+1)}$ , where a draw  $\mathbf{f}_{yes}^{(i)}$  from  $\mathcal{D}_{yes}^{(i)}$  is the random function  $\mathbf{f}_{yes}^{(i)}(x) = \mathbf{f}_{no}(x^{i \leftarrow 1})$  for all  $x \in \{0, 1\}^n$ , where  $\mathbf{f}_{no} \sim \mathcal{D}_{no}$  and  $x^{i \leftarrow 1}$  denotes the string  $x \in \{0, 1\}^{k+1}$  with its  $i$ -th bit set to 1. We see that  $\mathcal{D}_{yes}$  is supported entirely on  $k$ -juntas (in particular,  $\mathcal{D}_{yes}^{(i)}$  is supported entirely on functions that do not depend on the  $i$ -th coordinate), and a straightforward calculation shows  $\mathcal{D}_{no}$  is supported almost entirely on functions that are  $\Omega(\varepsilon)$ -far from being a  $k$ -junta:

► **Lemma 2.1** (Lemma 4.2 of [8]). *When  $6k/2^k < \varepsilon \leq 1/2$  and  $k \geq 3$ , a function  $\mathbf{f}_{no} : \{0, 1\}^{k+1} \rightarrow \{0, 1\}$  drawn from  $\mathcal{D}_{no}$  is  $(\varepsilon/6)$ -far from being a  $k$ -junta with probability at least  $11/12$ .*

We note that these functions  $\mathbf{f}_{yes}, \mathbf{f}_{no} : \{0, 1\}^{k+1} \rightarrow \{0, 1\}$  can be embedded in the full  $n$ -dimensional domain  $\{0, 1\}^n$  simply by defining  $\mathbf{F}_{yes} : \{0, 1\}^n \rightarrow \{0, 1\}$  where  $\mathbf{F}_{yes}(x) = \mathbf{f}_{yes}(x_{[k+1]})$  for all  $x \in \{0, 1\}^n$ , where  $x_{[k+1]}$  denotes the prefix substring  $(x_1, \dots, x_{k+1}) \in \{0, 1\}^{k+1}$  of  $x$ . Likewise, we may extend  $\mathbf{f}_{no} : \{0, 1\}^{k+1} \rightarrow \{0, 1\}$  to  $\mathbf{F}_{no} : \{0, 1\}^n \rightarrow \{0, 1\}$ . In the rest of the paper we confine our discussion to the  $\mathbf{f}_{yes}$  and  $\mathbf{f}_{no}$  functions over  $\{0, 1\}^{k+1}$ .

Fix any constant  $c < 1$ . Fix any query set  $Q^* = \{v^{(1)}, \dots, v^{(q)}\} \subseteq \{0, 1\}^{k+1}$  of cardinality  $q$  as specified in Equation (1) (we will specify the absolute constant  $C$  in Section 2.2 below). For now, we will let the ordering of the query strings  $v^{(1)}, \dots, v^{(q)}$  be arbitrary, though we will later impose a carefully chosen particular ordering (see Proposition 2.13). By a standard application of Yao’s minimax principle, to prove Theorem 1.1 it suffices to argue that  $d_{TV}(\mathbf{f}_{yes}(Q^*), \mathbf{f}_{no}(Q^*)) \leq 1/3$ , where  $\mathbf{f}_{yes}(Q^*)$  denotes the random “response vector”  $(\mathbf{f}_{yes}(v^{(1)}), \dots, \mathbf{f}_{yes}(v^{(q)})) \in \{0, 1\}^q$ , likewise  $\mathbf{f}_{no}(Q^*) = (\mathbf{f}_{no}(v^{(1)}), \dots, \mathbf{f}_{no}(v^{(q)})) \in \{0, 1\}^q$ , and  $d_{TV}(\cdot, \cdot)$  denotes the total variation distance (also known as statistical distance) between its two arguments.

**2.2 A useful Boolean isoperimetric inequality**

As discussed in the introduction, a key combinatorial lemma in Blais’  $\tilde{\Omega}(k/\varepsilon)$  sharpening of the Chockler–Gutfreund  $\Omega(k)$  lower bound is the classical edge-isoperimetric inequality of Harper, Bernstein, Lindsey, and Hart, which may be viewed as giving a lower bound on the cardinality of query sets in terms of the number of edges they induce.

► **Theorem 2.2** (Harper–Bernstein–Lindsey–Hart). *For all  $S \subseteq \{0, 1\}^n$ , we have  $|E_S| \leq \frac{1}{2}|S| \log |S|$ .*

We will need a variant of this inequality which takes into account the *directions* of the induced edges; in particular, it will be important for us that most directions have “not too few” induced edges in that direction.



► **Definition 2.3.** Let  $S \subseteq \{0, 1\}^n$ . We say that  $S$   $m$ -saturates direction  $i \in [n]$  if  $S$  induces at least  $m$  many  $i$ -edges.

Motivated by our earlier discussion in Section 1.3, a “good” query set  $Q \subseteq \{0, 1\}^{k+1}$  for distinguishing between  $\mathbf{f}_{yes} \sim \mathcal{D}_{yes}$  and  $\mathbf{f}_{no} \sim \mathcal{D}_{no}$  is one which  $m$ -saturates most of the  $k+1$  coordinates for a suitable choice of  $m$  (and of course we want  $Q$  to achieve this while being as small as possible). What kind of query sets  $Q$  are best suited to meet these two objectives? As an easy first observation, let  $Q_1$  be an arbitrary query set such that  $G_{Q_1}$  has  $q_{1,i}$  edges in each direction  $i$ . It is not difficult to show that there exists a query set  $Q_2$ , with  $|Q_2| = |Q_1|$ , such that (i)  $G_{Q_2}$  is a connected graph and (ii)  $G_{Q_2}$  has  $q_{2,i} \geq q_{1,i}$  edges in each direction  $i$ . (Repeatedly translate connected components of  $Q_1$  until they “come together” and only a single connected component is present; such translations cannot decrease the number of edges in any direction.)

In fact, a stronger statement than the above is true (and is not difficult to show): the “best” query set  $Q$  of a given size is of the form  $g^{-1}(1)$  (or  $g^{-1}(0)$ ) for some monotone Boolean function  $g$ . This is made precise through the following definition and fact:

► **Definition 2.4.** For each  $i \in [n]$  the  $i$ -th down-shift operator  $\kappa_i$  acts on Boolean functions  $g : \{0, 1\}^n \rightarrow \{0, 1\}$  as follows:  $(\kappa_i g)(x) = g(x)$  if  $g(x) = g(x^{\oplus i})$ , and  $(\kappa_i g)(x) = 1 - x_i$  otherwise.

► **Fact 2.5** (see e.g. [5]). Let  $S \subseteq \{0, 1\}^n$  and  $g : \{0, 1\}^n \rightarrow \{0, 1\}$  be its indicator function. Consider  $S_{\text{shift}} := g_{\text{shift}}^{-1}(1) \subseteq \{0, 1\}^n$ , where  $g_{\text{shift}} := \kappa_1 \cdots \kappa_n g$ . Then  $|S_{\text{shift}}| = |S|$  and  $S_{\text{shift}}$  is downward closed, meaning that for all  $v' \preceq v$ , if  $v \in S_{\text{shift}}$  then  $v' \in S_{\text{shift}}$ . Furthermore, if  $G_S$  has  $q_i$  edges in direction  $i$ , then  $G_{S_{\text{shift}}}$  has  $q_{\text{shift},i} \geq q_i$  edges in direction  $i$  (hence if  $S$   $m$ -saturates a direction  $i$  then so does  $S_{\text{shift}}$ ).

The following isoperimetric bound plays a key role in our arguments; it says that we need “many” vertices to  $m$ -saturate a large number of distinct directions.

► **Proposition 2.6.** Let  $S \subseteq \{0, 1\}^n$  be a set of points that  $m$ -saturates at least  $\ell$  directions. Then  $|S| \geq \frac{m\ell}{\lfloor \log m + 1 \rfloor} = \Omega\left(\frac{m\ell}{\log m}\right)$ .

**Proof.** Let  $\text{height}(S)$  denote the quantity  $\max_{v \in S} \|v\|$ , where  $\|v\| = \sum_{i=1}^n v_i$  is the Hamming weight of  $v \in \{0, 1\}^n$ . By Fact 2.5, we may restrict our attention to sets  $S$  that are downward closed. Let  $S^*$  be a downward-closed set of minimal size that  $m$ -saturates at least  $\ell$  directions, and which has  $\text{height}(S^*)$  as small as possible among all such minimal sets; for brevity we write  $h$  to denote  $\text{height}(S^*)$ . Note that we have the relationship

$$m\ell \leq |E_{S^*}| = \sum_{v \in S^*} \|v\| \leq h \cdot |S^*|, \quad (2)$$

and hence to prove a lower bound on the size of  $S^*$ , it suffices to show an upper bound on  $h$ , the height of  $S^*$ . Let  $v^*$  be a vertex in  $S$  with  $\|v^*\| = h$ , let  $D_{v^*} = \{i \in [n] : v_i^* = 1\}$ , and consider  $S' = S^* \setminus \{v^*\}$ . Since  $S^*$  is downward closed we have that  $G_{S^*}$  has at least  $2^{h-1}$  edges in each direction  $i \in D_{v^*}$ . Deleting  $v^*$  from  $S^*$  removes exactly  $h$  induced edges, one from each direction  $i \in D_{v^*}$ , and so by the minimality of  $S^*$  it follows that  $2^{h-1} - 1 < m$ , or equivalently,  $h \leq \lfloor \log m + 1 \rfloor$ . This, with (2), completes the proof. ◀

► **Remark.** Proposition 2.6 recovers as a special case a classical result of Frankl (Theorem 4 of [27]), proved using the Kruskal–Katona theorem, giving a lower bound of  $|S| = \Omega\left(\frac{m\ell}{\log m}\right)$  on the cardinality of any set  $S \subseteq \{0, 1\}^n$  which  $m$ -saturates all  $n$  directions. We note also



that the parameters of Proposition 2.6 are optimal up to a factor of 2. To see this, suppose  $t := \log m + 1 \in \mathbb{N}$  and  $t$  divides  $\ell$ . Let  $A_1, \dots, A_{\ell/t}$  be a partition of  $[\ell]$  into disjoint blocks of cardinality  $t$ , and for each  $i \in [\ell/t]$ , let  $C_i = \{v: v_j = 0 \text{ for all } j \notin A_i\}$  be the  $t$ -dimensional subcube over the coordinates in  $A_i \subseteq [\ell]$ . Then  $S := \bigcup C_i$  is a set of cardinality  $|S| = (2m\ell/(\log m + 1)) - 1$  which  $m$ -saturates the first  $\ell$  directions.

By Proposition 2.6, we can and shall assume that the query set  $Q^* \subseteq \{0, 1\}^{k+1}$  (which is of size at most  $q \leq \frac{Ck \log k}{\varepsilon^c \log((\log k)/\varepsilon^c)}$ , recall (1))  $(\log k)/\varepsilon^c$ -saturates at most  $0.1k$  directions. As noted earlier, by Yao's minimax principle, to prove Theorem 1.1 it remains to argue that  $d_{\text{TV}}(\mathbf{f}_{\text{yes}}(Q^*), \mathbf{f}_{\text{no}}(Q^*)) \leq 1/3$ .

### 2.3 Conditioning on unsaturated irrelevant coordinates, and bounding total variation by establishing concentration

In analyzing the random variable  $\mathbf{f}_{\text{yes}}(Q^*)$ , it will be helpful for us to condition on the event that  $Q^*$  only induces ‘‘a few’’ edges in the direction of the irrelevant coordinate  $i \in [k+1]$ . Formally, let  $U \subseteq [k+1]$  denote the directions that are not  $((\log k)/\varepsilon^c)$ -saturated by  $Q^*$ , and recall that  $|U| \geq 0.9k$  by our assumption on the cardinality of  $Q^*$  along with Proposition 2.6. Let  $\mathcal{D}'_{\text{yes}}$  denote the uniform mixture of  $\mathcal{D}_{\text{yes}}^{(i)}$  for all  $i \in U$  (i.e.  $\mathcal{D}'_{\text{yes}}$  is  $\mathcal{D}_{\text{yes}}$  conditioned on the irrelevant coordinate  $i$  being in  $U$ ), and  $\mathcal{D}''_{\text{yes}}$  denote the uniform mixture of  $\mathcal{D}_{\text{yes}}^{(i)}$  for all  $i \notin U$ . In other words,  $\mathcal{D}_{\text{yes}}$  is the mixture of  $\mathcal{D}'_{\text{yes}}$  and  $\mathcal{D}''_{\text{yes}}$  with mixing weights  $1 - \delta$  and  $\delta$  respectively, where  $\delta \leq 0.1$ . We write  $\mathbf{f}'_{\text{yes}}$  and  $\mathbf{f}''_{\text{yes}}$  to denote draws from  $\mathcal{D}'_{\text{yes}}$  and  $\mathcal{D}''_{\text{yes}}$  respectively.

► **Lemma 2.7.**  $d_{\text{TV}}(\mathbf{f}_{\text{yes}}(Q^*), \mathbf{f}_{\text{no}}(Q^*)) \leq d_{\text{TV}}(\mathbf{f}'_{\text{yes}}(Q^*), \mathbf{f}_{\text{no}}(Q^*)) + \delta$ .

**Proof.** This holds by noting that  $d_{\text{TV}}(\mathbf{f}_{\text{yes}}(Q^*), \mathbf{f}_{\text{no}}(Q^*))$  can be expressed as

$$\begin{aligned} & \frac{1}{2} \sum_{y \in \{0,1\}^q} |(1-\delta) \Pr[\mathbf{f}'_{\text{yes}}(Q^*) = y] + \delta \Pr[\mathbf{f}''_{\text{yes}}(Q^*) = y] - \Pr[\mathbf{f}_{\text{no}}(Q^*) = y]| \\ & \leq \frac{1}{2} \sum_{y \in \{0,1\}^q} |\Pr[\mathbf{f}'_{\text{yes}}(Q^*) = y] - \Pr[\mathbf{f}_{\text{no}}(Q^*) = y]| \\ & \quad + \delta (\Pr[\mathbf{f}'_{\text{yes}}(Q^*) = y] + \Pr[\mathbf{f}''_{\text{yes}}(Q^*) = y]) \\ & = d_{\text{TV}}(\mathbf{f}'_{\text{yes}}(Q^*), \mathbf{f}_{\text{no}}(Q^*)) + \delta. \end{aligned} \quad \blacktriangleleft$$

And so indeed, Lemma 2.7 reduces the task of proving  $d_{\text{TV}}(\mathbf{f}_{\text{yes}}(Q^*), \mathbf{f}_{\text{no}}(Q^*)) \leq 1/3$  to that of showing

$$d_{\text{TV}}(\mathbf{f}'_{\text{yes}}(Q^*), \mathbf{f}_{\text{no}}(Q^*)) \leq (1/3) - 0.1, \quad (3)$$

which is what we will do. We begin by observing that the distribution of  $\mathbf{f}_{\text{no}}(Q^*)$  is fairly easy to understand: for all  $y \in \{0, 1\}^q$ , we have  $\Pr[\mathbf{f}_{\text{no}}(Q^*) = y] = \varepsilon^{|y|} (1 - \varepsilon)^{q - |y|} := \text{wt}_\varepsilon(y)$ . This motivates us to define the function  $A: \{0, 1\}^{Q^*} \rightarrow [0, 1]$ ,

$$A(y) = \Pr[\mathbf{f}'_{\text{yes}}(Q^*) = y], \text{ and write } d_{\text{TV}}(\mathbf{f}'_{\text{yes}}(Q^*), \mathbf{f}_{\text{no}}(Q^*)) = \sum_{y \in \{0,1\}^{Q^*}} \frac{|A(y) - \text{wt}_\varepsilon(y)|}{2}.$$

For the remainder of this proof, we will write  $\mathbf{y} = (\mathbf{y}_1, \dots, \mathbf{y}_q)$  to denote a draw from  $\{0, 1\}_{(\varepsilon)}^q$ , the  $\varepsilon$ -biased product distribution over  $\{0, 1\}^q$  where each coordinate is independently 1 with probability  $\varepsilon$ . It will be convenient to think of  $\mathbf{y}$  as the values  $\mathbf{f}_{\text{no}} \sim \mathcal{D}_{\text{no}}$  takes on the query

points in  $Q^* \subseteq \{0, 1\}^{k+1}$ ; in other words,  $\mathbf{y}$  is distributed identically to  $\mathbf{f}_{no}(Q^*)$ . Writing  $\tilde{A}(\mathbf{y}) := A(\mathbf{y})/\text{wt}_\varepsilon(\mathbf{y})$ , we have

$$d_{\text{TV}}(\mathbf{f}'_{yes}(Q^*), \mathbf{f}_{no}(Q^*)) = \frac{1}{2} \sum_{\mathbf{y} \in \{0, 1\}^{Q^*}} \text{wt}_\varepsilon(\mathbf{y}) |\tilde{A}(\mathbf{y}) - 1| = \frac{1}{2} \mathbf{E} [|\tilde{A}(\mathbf{y}) - 1|].$$

Since  $1 = \sum_{\mathbf{y}} A(\mathbf{y}) = \sum_{\mathbf{y}} \text{wt}_\varepsilon(\mathbf{y}) \cdot \tilde{A}(\mathbf{y}) = \mathbf{E}[\tilde{A}(\mathbf{y})]$ , it suffices for us to argue that the random variable  $\tilde{A}(\mathbf{y})$  is concentrated around its expectation  $\mathbf{E}[\tilde{A}(\mathbf{y})] = 1$ :

► **Proposition 2.8.**  $\Pr [\tilde{A}(\mathbf{y}) \in [0.9, 1.1]] \geq 0.9$ .

Our claimed bound on total variation distance (3) follows from Proposition 2.8 via the following calculation, where the penultimate inequality uses Proposition 2.8:

$$\begin{aligned} d_{\text{TV}}(\mathbf{f}'_{yes}(Q^*), \mathbf{f}_{no}(Q^*)) &= \frac{1}{2} \sum_{\mathbf{y} \in \{0, 1\}^q} |A(\mathbf{y}) - \text{wt}_\varepsilon(\mathbf{y})| \\ &= \frac{1}{2} \left( 2 - 2 \sum_{\mathbf{y} \in \{0, 1\}^q} \min\{A(\mathbf{y}), \text{wt}_\varepsilon(\mathbf{y})\} \right) \\ &\leq 1 - \sum_{\substack{\mathbf{y} \in \{0, 1\}^q \\ \tilde{A}(\mathbf{y}) \in [0.9, 1.1]}} \min\{A(\mathbf{y}), \text{wt}_\varepsilon(\mathbf{y})\} \\ &\leq 1 - \sum_{\substack{\mathbf{y} \in \{0, 1\}^q \\ \tilde{A}(\mathbf{y}) \in [0.9, 1.1]}} 0.9 \cdot \text{wt}_\varepsilon(\mathbf{y}) \leq 1 - (0.9)^2 < (1/3) - 0.1. \end{aligned}$$

## 2.4 Proof of Proposition 2.8

We will bound the probability that  $\tilde{A}(\mathbf{y})$  deviates from its mean using the “method of averaged bounded differences” in which a rare “bad” event is allowed to take place:

► **Theorem 2.9** (special case of Theorem 3.7 of [36]). *Let  $\tilde{A}$  be a function of  $\{0, 1\}$ -valued random variables  $\mathbf{y}_1, \dots, \mathbf{y}_q$  such that  $\mathbf{E}[\tilde{A}(\mathbf{y})]$  is bounded. Let  $\mathcal{B} \subseteq \{0, 1\}^q$ , and suppose that for all  $b \in \{0, 1\}^q \setminus \mathcal{B}$ ,*

$$\sum_{j \in [q]} (\mathbf{E} [\tilde{A}(b_1, \dots, b_{j-1}, b_j, \mathbf{y}_{j+1}, \dots, \mathbf{y}_q) - \tilde{A}(b_1, \dots, b_{j-1}, \bar{b}_j, \mathbf{y}_{j+1}, \dots, \mathbf{y}_q)])^2 \leq \Delta. \quad (4)$$

Then for all  $t \geq 0$ , we have  $\Pr [|\tilde{A}(\mathbf{y}) - \mathbf{E}[\tilde{A}(\mathbf{y})]| > t] \leq 2 \exp(-2t^2/\Delta) + 2 \Pr[\mathbf{y} \in \mathcal{B}]$ .

We introduce some useful notation. Given a labelling  $b = (b_1, \dots, b_q) \in \{0, 1\}^q$  of the query strings  $v^{(1)}, \dots, v^{(q)}$  in  $Q^*$ , we write  $(b_j, \mathbf{y}_{j+1})$  to denote the hybrid string  $(b_1, \dots, b_{j-1}, b_j, \mathbf{y}_{j+1}, \dots, \mathbf{y}_q)$  and likewise  $(\bar{b}_j, \mathbf{y}_{j+1})$  to denote  $(b_1, \dots, b_{j-1}, \bar{b}_j, \mathbf{y}_{j+1}, \dots, \mathbf{y}_q)$ . We also write  $\text{diff}(b, j)$  to denote the difference  $|\mathbf{E}[\tilde{A}(b_j, \mathbf{y}_{j+1}) - \tilde{A}(\bar{b}_j, \mathbf{y}_{j+1})]|$ . This notational convention allows us to express the inequality (4) more succinctly as

$$\sum_{j \in [q]} \text{diff}(b, j)^2 = \sum_{j \in [q]} (\mathbf{E}[\tilde{A}(b_j, \mathbf{y}_{j+1}) - \tilde{A}(\bar{b}_j, \mathbf{y}_{j+1})])^2 \leq \Delta. \quad (5)$$

Furthermore, we write  $\#_i 11(b)$  to denote the number of  $i$ -edges in  $G_{Q^*}$  whose endpoints are both labeled 1 by  $b$ , and likewise  $\#_i 00(b)$  to denote the number of  $i$ -edges in  $G_{Q^*}$  whose endpoints are both labeled 0 by  $b$ . We write  $\#_i 1(b)$  to denote the number of vertices in  $G_{Q^*}$

that are labeled 1 by  $b$  and are not incident to an  $i$ -edge in  $G_{Q^*}$ , and likewise  $\#_i 0(b)$ . Finally, let  $i\text{-bias}_\varepsilon(b)$  denote the quantity  $\varepsilon^{-\#_i 1(b)} \cdot (1 - \varepsilon)^{-\#_i 0(b)}$ .

The following terminology will be useful: We say a labeling  $y \in \{0, 1\}^q$  of the query strings  $v^{(1)}, \dots, v^{(q)} \in Q^*$  is  $i$ -monochromatic (abbreviated as “ $i$ -mono”) if every  $i$ -edge in  $G_{Q^*}$  either has both endpoints labeled 1 by  $y$ , or has both endpoints labeled 0 by  $y$ . With this notation and terminology in place we may conveniently characterize  $\tilde{A}(y)$  as follows:

► **Lemma 2.10.**  $\tilde{A}(y) = \frac{1}{|U|} \sum_{i \in U} \mathbf{1}[y \text{ is } i\text{-mono}] \cdot i\text{-bias}_\varepsilon(y)$ .

**Proof of Lemma 2.10.** Fix a choice for the irrelevant coordinate  $i \in U$ . Conditioned on this, the probability that  $\mathbf{f}'_{y_{es}}(Q^*) = y$  is  $\varepsilon^{\#_i 1(y) + \#_i 1(y)} \cdot (1 - \varepsilon)^{\#_i 0(y) + \#_i 0(y)}$  if  $y$  is  $i$ -mono and is 0 otherwise. As  $\text{wt}_\varepsilon(y) = \varepsilon^{2\#_i 1(y) + \#_i 1(y)} \cdot (1 - \varepsilon)^{2\#_i 0(y) + \#_i 0(y)}$  if  $y$  is  $i$ -mono, we have that  $\tilde{A}(y)$  equals

$$\begin{aligned} & \frac{1}{|U|} \sum_{i \in U} \mathbf{1}[y \text{ is } i\text{-mono}] \cdot \frac{1}{\text{wt}_\varepsilon(y)} \cdot \Pr[\mathbf{f}'_{y_{es}}(Q^*) = y \mid \mathbf{i} = i] \\ &= \frac{1}{|U|} \sum_{i \in U} \mathbf{1}[y \text{ is } i\text{-mono}] \cdot \frac{\varepsilon^{\#_i 1(y) + \#_i 1(y)} \cdot (1 - \varepsilon)^{\#_i 0(y) + \#_i 0(y)}}{\varepsilon^{2\#_i 1(y) + \#_i 1(y)} \cdot (1 - \varepsilon)^{2\#_i 0(y) + \#_i 0(y)}} \\ &= \frac{1}{|U|} \sum_{i \in U} \mathbf{1}[y \text{ is } i\text{-mono}] \cdot i\text{-bias}_\varepsilon(y). \quad \blacktriangleleft \end{aligned}$$

By Lemma 2.10, we have that  $\text{diff}(b, j)$  is at most  $\frac{1}{|U|}$  times

$$\left| \underbrace{\sum_{i \in U} \mathbf{E}_y \left[ \mathbf{1}[(b_j, \mathbf{y}_{j+1}) \text{ is } i\text{-mono}] i\text{-bias}_\varepsilon(b_j, \mathbf{y}_{j+1}) - \mathbf{1}[(\bar{b}_j, \mathbf{y}_{j+1}) \text{ is } i\text{-mono}] i\text{-bias}_\varepsilon(\bar{b}_j, \mathbf{y}_{j+1}) \right]}_{(*)} \right|. \quad (6)$$

Fix  $b \in \{0, 1\}^{Q^*}$  and  $j \in [q]$ . We make a couple of observations about the quantity  $(*)$  for a fixed coordinate  $i \in U$  which will be useful later.

► **Observation 2.11.** *If  $v^{(j)}$  is not incident to an  $i$ -edge within  $G_{Q^*}$ , then  $(*) = 0$  pointwise for every possible outcome of  $\mathbf{y}$ .*

This is because the labeling of  $v^{(j)}$  has no effect on either the monochromaticity of the  $i$ -th direction or the number of monochromatic  $i$ -edges, and hence  $\mathbf{1}[(b_j, \mathbf{y}_{j+1}) \text{ is } i\text{-mono}] = \mathbf{1}[(\bar{b}_j, \mathbf{y}_{j+1}) \text{ is } i\text{-mono}]$  and  $i\text{-bias}_\varepsilon(b_j, \mathbf{y}_{j+1}) = i\text{-bias}_\varepsilon(\bar{b}_j, \mathbf{y}_{j+1})$  for every possible outcome  $\mathbf{y}$  of  $\mathbf{y}$ .

► **Observation 2.12.** *If  $v^{(j)}$  has an  $i$ -edge to  $v^{(j')}$  within  $G_{Q^*}$  where  $j' > j$ , then again  $(*) = 0$ .*

(In the following equations we use the notation  $(a_1, a_2, \mathbf{y}_{j+2})$  to denote the string  $(b_j, \mathbf{y}_{j+1})$  except with the  $j$ -th bit set to  $a_1$  and the  $j'$ -th bit set to  $a_2$ .) Observation 2.12 is true because

$$\begin{aligned} \pm(*) &= \mathbf{E}_y \left[ \mathbf{1}[(1, \mathbf{y}_{j+1}) \text{ is } i\text{-mono}] i\text{-bias}_\varepsilon(1, \mathbf{y}_{j+1}) - \mathbf{1}[(0, \mathbf{y}_{j+1}) \text{ is } i\text{-mono}] i\text{-bias}_\varepsilon(0, \mathbf{y}_{j+1}) \right] \\ &= \mathbf{E}_y \left[ \varepsilon \mathbf{1}[(1, 1, \mathbf{y}_{j+2}) \text{ is } i\text{-mono}] i\text{-bias}_\varepsilon(1, 1, \mathbf{y}_{j+2}) \right. \\ &\quad \left. - (1 - \varepsilon) \mathbf{1}[(0, 0, \mathbf{y}_{j+2}) \text{ is } i\text{-mono}] i\text{-bias}_\varepsilon(0, 0, \mathbf{y}_{j+2}) \right], \end{aligned}$$

and moreover,  $\mathbf{1}[(1, 1, \mathbf{y}_{j+2}) \text{ is } i\text{-mono}] = \mathbf{1}[(0, 0, \mathbf{y}_{j+2}) \text{ is } i\text{-mono}]$  and  $\varepsilon \cdot i\text{-bias}_\varepsilon(1, 1, \mathbf{y}_{j+2}) = (1 - \varepsilon) \cdot i\text{-bias}_\varepsilon(0, 0, \mathbf{y}_{j+2})$  for every possible outcome  $\mathbf{y}$  of  $\mathbf{y}$ .

### 2.4.1 Choosing an ordering

Given the preceding observations, we may rewrite (6) so that the sum is only over those directions  $i \in U$  such that  $v^{(j)}$  has an  $i$ -edge within  $G_{Q^*}$  to some  $v^{(j')}$  where  $j' < j$ . A priori, there is no reason to believe that this rewriting will simplify the sum or significantly reduce the number of summands. However, the next proposition shows that by enforcing an appropriate ordering on the query set  $Q^* = \{v^{(1)}, \dots, v^{(q)}\}$ , we can ensure that all but  $\lfloor \log(q) \rfloor$  terms will drop out of (6).

► **Proposition 2.13.** *For every  $S = \{v^{(1)}, \dots, v^{(q)}\} \subseteq \{0, 1\}^n$ , there exists an ordering  $v^{(1)} \prec v^{(2)} \prec \dots \prec v^{(q)}$  such that every  $v^{(i)}$  has at most  $\lfloor \log q \rfloor$  many Hamming neighbors  $v^{(j)}$  that precede it in the ordering.*

**Proof.** We proceed by induction on  $q$ , noting that the lemma trivially holds when  $q = 1$ . For the inductive step, we partition  $S$  into  $S_L$  and  $S_R$ , where

$$S_L = \{v \in S : \deg_S(v) \leq \lfloor \log q \rfloor\},$$

$$S_R = \{v \in S : \deg_S(v) > \lfloor \log q \rfloor\},$$

and  $\deg_S(v)$  denotes the degree of  $v$  in  $G_S$ . By the edge-isoperimetric inequality (Theorem 2.2), we have that

$$\sum_{v \in S} \deg_S(v) = 2 \cdot |E_S| \leq q \log q,$$

and hence  $|S_L| \geq 1$ , or equivalently,  $|S_R| \leq q - 1$ . By our induction hypothesis applied to  $S_R$ , there exists an ordering of its vertices so that every vertex has at most  $\lfloor \log |S_R| \rfloor \leq \lfloor \log q \rfloor$  Hamming neighbors that precede it in the ordering. Our ordering of  $S$  will be the vertices in  $S_R$  listed in this order given by the induction hypothesis, followed by the vertices in  $S_L$  listed in an arbitrary order. The proof is complete by recalling that  $\deg_S(v) \leq \lfloor \log q \rfloor$  for every  $v \in S_L$ , and hence every  $v \in S_L$  trivially has at most  $\lfloor \log q \rfloor$  Hamming neighbors that precede it in the ordering. ◀

We will now assume that  $Q^* = \{v^{(1)}, \dots, v^{(q)}\}$  is sorted in the order given by Proposition 2.13. Since  $|Q^*| = q = o(k^{1.1}) \ll k^2$  (recall (1) and the bounds on  $\varepsilon$  given in the conditions of Theorem 1.1) we have that there are fewer than  $2 \log k$  such directions  $i \in U$ . Let  $i^* \in U$  be the direction that maximizes (\*) in (6), and so  $\text{diff}(b, j)$  is at most (6), which in turn is at most  $\frac{2 \log k}{0.9k}$  times

$$\left| \mathbf{E}_{\mathbf{y}} \left[ \underbrace{\mathbf{1}[(b_j, \mathbf{y}_{j+1}) \text{ is } i^*\text{-mono}]}_{(**)} \cdot i^*\text{-bias}_{\varepsilon}(b_j, \mathbf{y}_{j+1}) - \underbrace{\mathbf{1}[(\bar{b}_j, \mathbf{y}_{j+1}) \text{ is } i^*\text{-mono}]}_{(***)} \cdot i^*\text{-bias}_{\varepsilon}(\bar{b}_j, \mathbf{y}_{j+1}) \right] \right|.$$

Since  $v^{(j)}$  has an  $i^*$ -edge within  $G_{Q^*}$  to some  $v^{(j')}$  where  $j' < j$ , it follows that either  $\mathbf{E}[(**)] = 0$  or  $\mathbf{E}[(***)] = 0$  (the former if  $b_{j'} \neq b_j$ , and the latter if  $b_{j'} \neq \bar{b}_j$ ). We may assume w.l.o.g. that  $\mathbf{E}[(***)] = 0$ , and so

$$\text{diff}(b, j) \leq \frac{2 \log k}{0.9k} \mathbf{E}_{\mathbf{y}} \left[ \mathbf{1}[(b_j, \mathbf{y}_{j+1}) \text{ is } i^*\text{-mono}] \cdot i^*\text{-bias}_{\varepsilon}(b_j, \mathbf{y}_{j+1}) \right].$$

Next, we observe that the expectation above may be rewritten as

$$\begin{aligned} & \mathbf{E}_{\mathbf{y}} \left[ \mathbf{1}[(b_j, \mathbf{y}_{j+1}) \text{ is } i^*\text{-mono}] \cdot i^*\text{-bias}_{\varepsilon}(b_j, \mathbf{y}_{j+1}) \right] \\ &= \prod_{i^*\text{-edges } e} \mathbf{E}_{\mathbf{y}} \left[ \mathbf{1}[e \text{ is mono w.r.t. } (b_j, \mathbf{y}_{j+1})] \cdot i^*\text{-bias}_{\varepsilon}((b_j, \mathbf{y}_{j+1})|_e) \right], \end{aligned} \quad (7)$$

where

$$i^*\text{-bias}_\varepsilon((b_j, \mathbf{y}_{j+1})|_e) = \begin{cases} \varepsilon^{-1} & \text{if both endpoints of } e \text{ are labeled 1 by } (b_j, \mathbf{y}_{j+1}) \\ (1 - \varepsilon)^{-1} & \text{if both endpoints of } e \text{ are labeled 0 by } (b_j, \mathbf{y}_{j+1}). \end{cases}$$

We claim that the expectation on the RHS of (7) is 1 unless  $e = (v^{(\ell)}, v^{(r)})$ , where  $\ell < r \leq j$ . To see this, note that if  $j < \ell < r$  then

$$\mathbf{E}_{\mathbf{y}} \left[ \mathbf{1}[e \text{ is mono w.r.t. } (b_j, \mathbf{y}_{j+1})] \cdot i^*\text{-bias}_\varepsilon((b_j, \mathbf{y}_{j+1})|_e) \right] = \varepsilon^2 \cdot \frac{1}{\varepsilon} + (1 - \varepsilon)^2 \cdot \frac{1}{1 - \varepsilon} = 1,$$

and if  $\ell \leq j \leq r$  then

$$\mathbf{E}_{\mathbf{y}} \left[ \mathbf{1}[e \text{ is mono w.r.t. } (b_j, \mathbf{y}_{j+1})] \cdot i^*\text{-bias}_\varepsilon((b_j, \mathbf{y}_{j+1})|_e) \right] = \begin{cases} \varepsilon \cdot \varepsilon^{-1} & \text{if } b_\ell = 1 \\ (1 - \varepsilon) \cdot (1 - \varepsilon)^{-1} & \text{if } b_\ell = 0. \end{cases}$$

Since neither  $\mathbf{1}[e \text{ is mono w.r.t. } (b_j, \mathbf{y}_{j+1})]$  nor  $i^*\text{-bias}_\varepsilon((b_j, \mathbf{y}_{j+1})|_e)$  depend on  $\mathbf{y}$  when  $e = (v^{(\ell)}, v^{(r)})$  where  $\ell < r \leq j$ , it follows that (7) may be simplified to be

$$(7) = \prod_{\substack{i^*\text{-edges } e \\ e=(v^{(\ell)}, v^{(r)}), \ell < r \leq j}} \mathbf{1}[e \text{ is mono w.r.t. } b] \cdot i^*\text{-bias}_\varepsilon(b|_e).$$

Recalling that this expression (7) depends on both  $b \in \{0, 1\}^q$  and  $j \in [q]$  (since  $i^*$  depends on  $j$ ), we write  $\text{val}(b, j)$  to denote (7), i.e.

$$\text{val}(b, j) := \prod_{\substack{i^*\text{-edges } e \\ e=(v^{(\ell)}, v^{(r)}), \ell < r \leq j}} \mathbf{1}[e \text{ is mono w.r.t. } b] \cdot i^*\text{-bias}_\varepsilon(b|_e) \tag{8}$$

and hence we may write

$$\text{diff}(b, j) \leq \frac{2 \log k}{0.9k} \cdot \text{val}(b, j).$$

### 2.4.2 Bounding $\text{val}(b, j)$ by bucketing

Our goal is to define a bad set  $\mathcal{B} \subseteq \{0, 1\}^q$  of small measure ( $\Pr[\mathbf{y} \in \mathcal{B}] \leq 0.01$  is sufficient, though our  $\mathcal{B}$  will satisfy  $\Pr[\mathbf{y} \in \mathcal{B}] = k^{-\Omega(1)}$ ) such that for all  $b \notin \mathcal{B}$ ,

$$\sum_{j \in [q]} \text{val}(b, j)^2 = O(k^{(\frac{25+c}{13})}). \tag{9}$$

This is sufficient since it implies that we may take  $\Delta := 0.01$  and have that the LHS of (4) is at most

$$\sum_{j \in [q]} \text{diff}(b, j)^2 \leq \left( \frac{2 \log k}{0.9k} \right)^2 \sum_{j \in [q]} \text{val}(b, j)^2 = \frac{1}{k^{2-o(1)}} \cdot O(k^{(\frac{25+c}{13})}) \leq \Delta = 0.01$$

for sufficiently large  $k$ . (This uses (5) along with the fact that  $c < 1$ .) Applying Theorem 2.9 with  $t = 0.1$  would then complete the proof of Proposition 2.8, and hence Theorem 1.1.

To reason about  $b \in \{0, 1\}^q$  for which (9) does not hold, we group the  $q$  many summands on the LHS of (9) into  $O(\log k)$  groups according to magnitude. Set  $M := (\frac{23+c}{24}) \log k$ ,

and partition  $[0, \infty)$  into  $M + 2$  intervals  $I_0 = [0, 1)$ ,  $I_m = [2^{m-1}, 2^m)$  for all  $m \in [M]$  and  $I_{M+1} = [2^M, \infty)$ . For each  $b \in \{0, 1\}^q$  and  $m \in \{0, 1, \dots, M + 1\}$  we define

$$\begin{aligned} \text{bucket}(b, m) &:= \{j \in [q] : \text{val}(b, j) \in I_m\}, \\ C(b, m) &:= \sum_{j \in \text{bucket}(b, m)} \text{val}(b, j)^2. \end{aligned}$$

With this notation in hand, we may write

$$\sum_{j \in [q]} \text{val}(b, j)^2 = \sum_{m=0}^{M+1} C(b, m). \quad (10)$$

Next, for each  $m \in \{0, 1, \dots, M + 1\}$  we define  $\mathcal{B}_m \subseteq \{0, 1\}^q$  to be

$$\mathcal{B}_m := \{b \in \{0, 1\}^q : C(b, m) > k^{\left(\frac{23+c}{12}\right)}\},$$

and finally  $\mathcal{B} := \bigcup \mathcal{B}_m$ . Certainly if  $b \notin \mathcal{B}$  then by (10) we have that

$$\sum_{j \in [q]} \text{val}(b, j)^2 = \sum_{m=0}^{M+1} C(b, m) \leq (M + 2) \cdot k^{\left(\frac{23+c}{12}\right)} = o(k^{\left(\frac{25+c}{13}\right)}),$$

and so it suffices to prove the following proposition.

► **Proposition 2.14.** *For all  $m \in \{0, 1, \dots, M + 1\}$ , we have that  $\Pr_{\mathbf{b} \sim \{0, 1\}_{(\varepsilon)}^q} [\mathbf{b} \in \mathcal{B}_m] = k^{-\Omega(1)}$ . (Consequently,  $\Pr[\mathbf{b} \in \mathcal{B}] = k^{-\Omega(1)}$  by a union bound.)*

**Proof.** First note that for all  $m \in \{0, 1, \dots, M\}$ , we have that

$$C(b, m) = \sum_{j \in \text{bucket}(b, m)} \text{val}(b, j)^2 < |\text{bucket}(b, m)| \cdot 2^{2m}. \quad (11)$$

Since  $|\text{bucket}(b, m)| \leq q = o(k^{1.1})$  for all  $m$ , we have that (11)  $< k^{1.7}$  for  $m \leq 0.3 \log k$ , and hence recalling the definition of  $\mathcal{B}_m$ , we have  $\Pr[\mathbf{b} \in \mathcal{B}_m] = 0$  for  $m \leq 0.3 \log k$ , so the proposition clearly holds for all such  $m$ . Hence we may assume that  $m > 0.3 \log k$ ; it will be convenient for us to write  $m = \alpha \log k$  for some  $\alpha \in (0.3, 1)$ . Next, observe that in order for  $C(b, m)$  to be greater than  $k^{\left(\frac{23+c}{12}\right)}$  it has to be the case that  $|\text{bucket}(b, m)| \geq k^{\left(\frac{23+c}{12}\right)} \cdot 2^{-2m} = k^{\left(\frac{23+c}{12}\right) - 2\alpha}$ ; this is trivially true for  $m = M + 1$  (since  $k^{\left(\frac{23+c}{12}\right)} \cdot 2^{-2m} = \frac{1}{4}$ ), and follows from (11) for  $m \in \{0, 1, \dots, M\}$ . We will therefore focus on bounding the RHS of

$$\Pr [|\text{bucket}(\mathbf{b}, m)| \geq k^{\left(\frac{23+c}{12}\right) - 2\alpha}] \leq \Pr [|\{j \in [q] : \text{val}(\mathbf{b}, j) \geq 2^{m-1} = \frac{1}{2}k^\alpha\}| \geq k^{\left(\frac{23+c}{12}\right) - 2\alpha}]$$

by  $k^{-\Omega(1)}$ . Consider the random variable  $\text{val}(\mathbf{b}, j)$  for a fixed  $j \in [q]$ . Let  $E = E(j)$  denote the number of  $i^*$ -edges (where  $i^*$  depends on  $j$ ), and note that  $E \leq \log(k)/\varepsilon^c$  since  $i^* \in U$  is an unsaturated direction. Recalling (8), we may introduce independent random variables  $\mathbf{X}_1^{(j)}, \dots, \mathbf{X}_E^{(j)}$  where

$$\mathbf{X}_\ell^{(j)} = \begin{cases} 0 & \text{with probability } 2\varepsilon(1 - \varepsilon) \\ (1 - \varepsilon)^{-1} & \text{with probability } (1 - \varepsilon)^2 \\ \varepsilon^{-1} & \text{with probability } \varepsilon^2 \end{cases}$$

and note that  $\text{val}(\mathbf{b}, j)$  (where  $\mathbf{b} \sim \{0, 1\}_{(\varepsilon)}^q$ ) is distributed identically to  $\prod_{\ell=1}^E \mathbf{X}_\ell^{(j)}$ . Simplifying further, we introduce additional (mutually independent) random variables  $\mathbf{Y}_1^{(j)}, \dots, \mathbf{Y}_E^{(j)}$ , where each  $\mathbf{Y}_\ell^{(j)}$  is coupled to  $\mathbf{X}_\ell^{(j)}$  in the following way

$$\mathbf{Y}_\ell^{(j)} = \begin{cases} \mathbf{X}_\ell^{(j)} & \text{when } \mathbf{X}_\ell^{(j)} = \varepsilon^{-1} \\ 1 & \text{otherwise.} \end{cases}$$

Under such a coupling, we have that

$$\prod_{\ell=1}^E \mathbf{X}_\ell^{(j)} \leq \left(\frac{1}{1-\varepsilon}\right)^E \prod_{\ell=1}^E \mathbf{Y}_\ell^{(j)}$$

with probability 1, where the factor  $(1-\varepsilon)^{-E}$  is  $\leq k^{\nu(k)}$  for some function  $\nu(k) = o_k(1)$ , for all  $\varepsilon = o_k(1)$  since  $E \leq \log(k)/\varepsilon^c$  (recall the bounds on  $\varepsilon$  given in the conditions of Theorem 1.1).

► **Claim 2.15.**  $\Pr \left[ \prod_{\ell=1}^E \mathbf{Y}_\ell^{(j)} \geq \frac{1}{2} k^{\alpha-\nu(k)} \right] = O(k^{-(\frac{4-c}{3})\alpha})$ .

**Proof of Claim 2.15.** Set  $t := (m - \nu(k) \log(k) - 2)/\log(1/\varepsilon) = ((\alpha - \nu(k)) \log(k) - 2)/\log(1/\varepsilon)$ , and so  $\varepsilon^{-t} = \frac{1}{2} k^{\alpha-\nu(k)}$ .

$$\begin{aligned} \Pr \left[ \prod_{\ell=1}^E \mathbf{Y}_\ell^{(j)} \geq \frac{k^{\alpha-\nu(k)}}{2} \right] &< \varepsilon^{2t} \binom{E}{t} \\ &\leq \frac{4}{k^{2\alpha-\nu(k)}} \left(\frac{eE}{t}\right)^t \\ &\leq \frac{4}{k^{2\alpha-\nu(k)}} \left(\frac{e \log(1/\varepsilon)}{0.3 \cdot \varepsilon^c}\right)^t \\ &\leq \frac{4}{k^{2\alpha-\nu(k)}} \cdot O\left(\varepsilon^{-\left(\frac{1+c}{2}\right)t}\right) \\ &< \frac{4}{k^{2\alpha-\nu(k)}} \cdot O\left(k^{\left(\frac{1+c}{2}\right)(\alpha-\nu(k))}\right) = O\left(\frac{1}{k^{\left(\frac{4-c}{3}\right)\alpha}}\right), \end{aligned}$$

where the third inequality uses the fact that  $t > 0.3 \log(k)/\log(1/\varepsilon)$  (recall our assumption that  $\alpha > 0.3$ ), the fourth inequality uses the fact that  $\varepsilon = o_k(1)$ , and the last inequality uses the fact that  $\nu(k) = o_k(1)$ . ◀

By linearity of expectation, it follows that

$$\mathbf{E} \left[ |\{j \in [q] : \text{val}(\mathbf{b}, j) \geq \frac{1}{2} k^\alpha\}| \right] = O(q \cdot k^{-(\frac{4-c}{3})\alpha}) = O(k^{(\frac{7-c}{6})} \cdot k^{-(\frac{4-c}{3})\alpha}) = O(k^{(\frac{7-c}{6}) - (\frac{4-c}{3})\alpha}),$$

where we have used the fact that  $q = O(k^{1+\eta})$  (recall (1) and the bounds on  $\varepsilon$  given in the conditions of Theorem 1.1) for any fixed  $\eta > 0$ , and so by Markov's inequality, we conclude that

$$\Pr \left[ |\{j \in [q] : \text{val}(\mathbf{b}, j) \geq \frac{1}{2} k^\alpha\}| \geq k^{(\frac{23+c}{12}) - 2\alpha} \right] = O(k^{(\frac{7-c}{6}) - (\frac{4-c}{3})\alpha - ((\frac{23+c}{12}) - 2\alpha)}) = O(k^{(\frac{c-1}{12})})$$

for sufficiently large  $k$ . Because  $c < 1$ , this is  $k^{-\Omega(1)}$ , and therefore the proof of Proposition 2.14 is complete. ◀

**Acknowledgements.** RS and LYT are supported by NSF grants CCF-1115703 and CCF-1319788, and JW is supported by NSF grants CCF-1115703, CCF-1319788, CCF-0747250, and CCF-1116594, and a Simons Fellowship in Theoretical Computer Science. Some of this work was done while LYT was at MSR-SVC and at Columbia University, and while JW was visiting Columbia University.

## References

- 1 Noga Alon, Eric Blais, Sourav Chakraborty, David García-Soriano, and Arie Matsliah. Nearly tight bounds for testing function isomorphism. *SIAM J. Comput.*, 42(2):459–493, 2013.
- 2 J. Arpe and E. Mossel. Application of a generalization of Russo’s formula to learning from multiple random oracles. *Combinatorics, Probability and Computing*, 19:183–199, 2010.
- 3 J. Arpe and R. Reischuk. Robust inference of relevant attributes. In *Proceedings of the Fourteenth International Conference on Algorithmic Learning Theory*, pages 99–113, 2003.
- 4 A. Atıçı and R. Servedio. Quantum algorithms for testing and learning juntas. *Quantum Information Processing*, 6(5):323–348, 2007.
- 5 M. Ben-Or and N. Linial. Collective coin flipping. In S. Micali, editor, *Randomness and Computation*, pages 91–115. Academic Press, 1990.
- 6 Arthur J. Bernstein. Maximally connected arrays on the  $n$ -cube. *SIAM J. Appl. Math.*, 15(6):1485–1489, 1967.
- 7 E. Blais. Testing juntas: A brief survey. In *Property Testing - Current Research and Surveys*, pages 32–40, 2010.
- 8 Eric Blais. Improved bounds for testing juntas. In *Proc. RANDOM*, pages 317–330, 2008.
- 9 Eric Blais. Testing juntas nearly optimally. In *Proceedings of STOC*, pages 151–158, 2009.
- 10 Eric Blais, Joshua Brody, and Badih Ghazi. The information complexity of hamming distance. In *RANDOM*, pages 462–486, 2014.
- 11 Eric Blais, Joshua Brody, and Kevin Matulef. Property testing lower bounds via communication complexity. In *CCC*, pages 210–220, 2011.
- 12 Eric Blais and Daniel M. Kane. Tight bounds for testing  $k$ -linearity. In *RANDOM*, pages 435–446, 2012.
- 13 Eric Blais and Ryan O’Donnell. Lower bounds for testing function isomorphism. In *IEEE Conference on Computational Complexity*, pages 235–246, 2010.
- 14 Eric Blais, Amit Weinstein, and Yuichi Yoshida. Partially symmetric functions are efficiently isomorphism-testable. In *FOCS*, pages 551–560, 2012.
- 15 A. Blum. Relevant examples and relevant features: Thoughts from computational learning theory. in *AAAI Fall Symposium on ‘Relevance’*, 1994.
- 16 A. Blum and P. Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97(1-2):245–271, 1997.
- 17 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.
- 18 Sourav Chakraborty, Eldar Fischer, David García-Soriano, and Arie Matsliah. Juntosymmetric functions, hypergraph isomorphism and crunching. In *CCC*, pages 148–158, 2012.
- 19 Sourav Chakraborty, David García-Soriano, and Arie Matsliah. Efficient sample extractors for juntas with applications. In *ICALP*, pages 545–556. Springer, 2011.
- 20 H. Chockler and D. Gutfreund. A lower bound for testing juntas. *Information Processing Letters*, 90(6):301–305, 2004.
- 21 D. Dachman-Soled, V. Feldman, L.-Y. Tan, A. Wan, and K. Wimmer. Approximate resilience, monotonicity, and the complexity of agnostic learning. In *SODA*, pages 498–511, 2015.
- 22 A. Dhagat and L. Hellerstein. PAC learning with irrelevant attributes. In *Proceedings of the Thirty-Fifth Annual Symposium on Foundations of Computer Science*, pages 64–74, 1994.
- 23 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.



- 24 I. Diakonikolas, H.K. Lee, K. Matulef, R. Servedio, and A. Wan. Efficiently testing sparse  $\text{GF}(2)$  polynomials. *Algorithmica*, 61(3):580–605, 2011.
- 25 Vitaly Feldman, Parikshit Gopalan, Subhash Khot, and Ashok Kumar Ponnuswami. On agnostic learning of parities, monomials, and halfspaces. *SIAM J. Comput.*, 39(2):606–645, 2009.
- 26 E. Fischer, G. Kindler, D. Ron, S. Safra, and A. Samorodnitsky. Testing juntas. *J. Computer & System Sciences*, 68(4):753–787, 2004.
- 27 Peter Frankl. On the trace of finite sets. *J. Comb. Theory, Ser. A*, 34(1):41–45, 1983.
- 28 O. Goldreich, editor. *Property Testing: Current Research and Surveys*. Springer, 2010. LNCS 6390.
- 29 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.
- 30 D. Guijarro, J. Tarui, and T. Tsukiji. Finding relevant variables in the PAC model with membership queries. In *Proceedings of the Tenth International Conference on Algorithmic Learning Theory*, pages 313–322, 1999.
- 31 Larry H. Harper. Optimal assignments of numbers to vertices. *SIAM J. Appl. Math.*, 12(1):131–135, 1964.
- 32 Sergiu Hart. A note on the edges of the  $n$ -cube. *Disc. Math.*, 14:157–163, 1976.
- 33 J. H. Lindsey. Assignment of numbers to vertices. *Amer. Math. Monthly*, 71:508–516, 1964.
- 34 K. Matulef, R. O’Donnell, R. Rubinfeld, and R. Servedio. Testing halfspaces. *SIAM J. on Comput.*, 39(5):2004–2047, 2010.
- 35 Kevin Matulef, Ryan O’Donnell, Ronitt Rubinfeld, and Rocco A. Servedio. Testing  $\pm 1$ -weight halfspace. In *APPROX-RANDOM*, pages 646–657, 2009.
- 36 Colin McDiarmid. Concentration. In *Probabilistic Methods for Algorithmic Discrete Mathematics*, pages 195–248, 1998.
- 37 E. Mossel, R. O’Donnell, and R. Servedio. Learning functions of  $k$  relevant variables. *Journal of Computer & System Sciences*, 69(3):421–434, 2004. Preliminary version in *Proc. STOC’03*.
- 38 D. Ron. Property Testing: A Learning Theory Perspective. *Foundations and Trends in Machine Learning*, 1(3):307–402, 2008.
- 39 D. Ron. Algorithmic and analysis techniques in property testing. *Foundations and Trends in Theoretical Computer Science*, 5:73–205, 2010.
- 40 D. Ron and R. Servedio. Exponentially improved algorithms and lower bounds for testing signed majorities. In *SODA*, pages 1319–1336, 2013.
- 41 D. Ron and G. Tsur. Testing computability by width-two OBDDs. *Theoretical Computer Science*, 420:64–79, 2012.
- 42 Gregory Valiant. Finding Correlations in Subquadratic Time, with Applications to Learning Parities and Juntas. In *FOCS*, pages 11–20, 2012.

# A Characterization of Hard-to-cover CSPs

Amey Bhangale<sup>\*1</sup>, Prahladh Harsha<sup>†2</sup>, and Girish Varma<sup>‡2</sup>

1 Department of Computer Science, Rutgers University, USA  
amey.bhangale@rutgers.edu

2 Tata Institute of Fundamental Research, India  
{prahladh,girishrv}@tifrr.res.in

---

## Abstract

We continue the study of *covering complexity* of constraint satisfaction problems (CSPs) initiated by Guruswami, Håstad and Sudan [9] and Dinur and Kol [7]. The *covering number* of a CSP instance  $\Phi$ , denoted by  $\nu(\Phi)$  is the smallest number of assignments to the variables of  $\Phi$ , such that each constraint of  $\Phi$  is satisfied by at least one of the assignments. We show the following results regarding how well efficient algorithms can approximate the covering number of a given CSP instance.

1. Assuming a *covering unique games conjecture*, introduced by Dinur and Kol, we show that for every non-odd predicate  $P$  over any constant sized alphabet and every integer  $K$ , it is NP-hard to distinguish between  $P$ -CSP instances (i.e., CSP instances where all the constraints are of type  $P$ ) which are coverable by a constant number of assignments and those whose covering number is at least  $K$ . Previously, Dinur and Kol, using the same covering unique games conjecture, had shown a similar hardness result for every non-odd predicate over the Boolean alphabet that supports a pairwise independent distribution. Our generalization yields a complete characterization of CSPs over constant sized alphabet  $\Sigma$  that are hard to cover since CSPs over odd predicates are trivially coverable with  $|\Sigma|$  assignments.
2. For a large class of predicates that are contained in the  $2k$ -LIN predicate, we show that it is quasi-NP-hard to distinguish between instances which have covering number at most two and covering number at least  $\Omega(\log \log n)$ . This generalizes the 4-LIN result of Dinur and Kol that states it is quasi-NP-hard to distinguish between 4-LIN-CSP instances which have covering number at most two and covering number at least  $\Omega(\log \log \log n)$ .

**1998 ACM Subject Classification** F.2 [Theory of Computation] Analysis of Algorithms and Problem Complexity

**Keywords and phrases** CSPs, Covering Problem, Hardness of Approximation, Unique Games, Invariance Principle

**Digital Object Identifier** 10.4230/LIPIcs.CCC.2015.280

## 1 Introduction

One of the central (yet unresolved) questions in inapproximability is the problem of coloring a (hyper)graph with as few colors as possible. A (hyper)graph  $G = (V, E)$  is said to be  $k$ -colorable if there exists a coloring  $c : V \rightarrow [k] := \{0, 1, 2, \dots, k-1\}$  of the vertices such that no (hyper)edge of  $G$  is monochromatic. The chromatic number of a (hyper)graph, denoted

---

\* Amey Bhangale's research is supported by the NSF grant CCF-1253886. Part of the work was done when the author was visiting TIFR.

† Prahladh Harsha's research is supported in part by the ISF-UGC grant number 1399/14.

‡ Girish Varma's research is supported by Google Ph.D. Fellowship in Algorithms.



by  $\chi(G)$ , is the smallest  $k$  such that  $G$  is  $k$ -colorable. It is known that computing  $\chi(G)$  to within a multiplicative factor of  $n^{1-\varepsilon}$  on an  $n$ -sized graph  $G$  for every  $\varepsilon \in (0, 1)$  is NP-hard. However, the complexity of the following problem is not yet completely understood: given a constant-colorable (hyper)graph, what is the minimum number of colors required to color the vertices of the graph efficiently such that every edge is non-monochromatic? The current best approximation algorithms for this problem require at least  $n^{\Omega(1)}$  colors while the hardness results are far from proving optimality of these approximation algorithms (see § 1.3 for a discussion on recent work in this area).

The notion of *covering complexity* was introduced by Guruswami, Håstad and Sudan [9] and more formally by Dinur and Kol [7] to obtain a better understanding of the complexity of this problem. Let  $P$  be a predicate and  $\Phi$  an instance of a constraint satisfaction problem (CSP) over  $n$  variables, where each constraint in  $\Phi$  is a constraint of type  $P$  over the  $n$  variables and their negations. We will refer to such CSPs as  $P$ -CSPs. The *covering number* of  $\Phi$ , denoted by  $\nu(\Phi)$ , is the smallest number of assignments to the variables such that each constraint of  $\Phi$  is satisfied by at least one of the assignments, in which case we say that the set of assignments *covers* the instance  $\Phi$ . If  $c$  assignments cover the instance  $\Phi$ , we say that  $\Phi$  is  $c$ -coverable or equivalently that the set of assignments form a  $c$ -covering for  $\Phi$ . The covering number is a generalization of the notion of chromatic number (to be more precise, the logarithm of the chromatic number) to all predicates in the following sense. Suppose  $P$  is the not-all-equal predicate NAE and the instance  $\Phi$  has no negations in any of its constraints, then the covering number  $\nu(\Phi)$  is exactly  $\lceil \log \chi(G_\Phi) \rceil$  where  $G_\Phi$  is the underlying constraint graph of the instance  $\Phi$ .

Cover- $P$  refers to the problem of finding the covering number of a given  $P$ -CSP instance. Finding the exact covering number for most interesting predicates  $P$  is NP-hard. We therefore study the problem of approximating the covering number. In particular, we would like to study the complexity of the following problem, denoted by COVERING- $P$ -CSP( $c, s$ ), for some  $1 \leq c < s \in \mathbb{N}$ : “given a  $c$ -coverable  $P$ -CSP instance  $\Phi$ , find an  $s$ -covering for  $\Phi$ ”. Similar problems have been studied for the Max-CSP setting: “for  $0 < s < c \leq 1$ , “given a  $c$ -satisfiable  $P$ -CSP instance  $\Phi$ , find an  $s$ -satisfying assignment for  $\Phi$ ”. Max-CSPs and Cover-CSPs, as observed by Dinur and Kol [7], are very different problems. For instance, if  $P$  is an odd predicate, i.e, if for every assignment  $x$ , either  $x$  or its negation  $x + \bar{1}$  satisfies  $P$ , then any  $P$ -CSP instance  $\Phi$  has a trivial two covering, any assignment and its negation. Thus, 3-LIN and 3-CNF<sup>1</sup>, being odd predicates, are easy to cover though they are hard predicates in the Max-CSP setting. The main result of Dinur and Kol is that the 4-LIN predicate, in contrast to the above, is hard to cover: for every constant  $t \geq 2$ , COVERING-4-LIN-CSP( $2, t$ ) is NP-hard. In fact, their arguments show that COVERING-4-LIN-CSP( $2, \Omega(\log \log \log n)$ ) is quasi-NP-hard.

Having observed that odd predicate based CSPs are easy to cover, Dinur and Kol proceeded to ask the question “are all non-odd-predicate CSPs hard to cover?”. In a partial answer to this question, they showed that assuming a covering variant of the unique games conjecture COVERING-UGC( $c$ ), if a predicate  $P$  is not odd and there is a balanced pairwise independent distribution on its support, then for all constants  $k$ , COVERING- $P$ -CSP( $2c, k$ ) is NP-hard (here,  $c$  is a fixed constant that depends on the covering variant of the unique games conjecture COVERING-UGC( $c$ )). See § 2 for the exact definition of the covering variant of the unique games conjecture.

<sup>1</sup> 3-LIN :  $\{0, 1\}^3 \rightarrow \{0, 1\}$  refers to the 3-bit predicate defined by  $3\text{-LIN}(x_1, x_2, x_3) := x_1 \oplus x_2 \oplus x_3$  while 3-CNF :  $\{0, 1\}^3 \rightarrow \{0, 1\}$  refers to the 3-bit predicate defined by  $3\text{-CNF}(x_1, x_2, x_3) := x_1 \vee x_2 \vee x_3$ .

### 1.1 Our Results

Our first result states that assuming the same covering variant of unique games conjecture COVERING-UGC( $c$ ) of Dinur and Kol [7], one can in fact show the covering hardness of *all* non-odd predicates  $P$  over *any* constant-sized alphabet  $[q]$ . The notion of odd predicate can be extended to any alphabet in the following natural way: a predicate  $P \subseteq [q]^k$  is odd if for all assignments  $x \in [q]^k$ , there exists  $a \in [q]$  such that the assignment  $x + \bar{a}$  satisfies  $P$ .

► **Theorem 1.1** (Covering hardness of non-odd predicates). *Assuming COVERING-UGC( $c$ ), for any constant-sized alphabet  $[q]$ , any constant  $k \in \mathbb{N}$  and any non-odd predicate  $P \subseteq [q]^k$ , for all constants  $t \in \mathbb{N}$ , the COVERING- $P$ -CSP( $2cq, t$ ) problem is NP-hard.*

Since odd predicates  $P \subseteq [q]^k$  are trivially coverable with  $q$  assignments, the above theorem, gives a *full characterization of hard-to-cover predicates* over any constant sized alphabet (modulo the covering variant of the unique games conjecture): a predicate is hard to cover iff it is not odd.

We then ask if we can prove similar covering hardness results under more standard complexity assumptions (such as  $\text{NP} \neq \text{P}$  or the exponential-time hypothesis (ETH)). Though we are not able to prove that every non-odd predicate is hard under these assumptions, we give sufficient conditions on the predicate  $P$  for the corresponding approximate covering problem to be quasi-NP-hard. Recall that  $2k\text{-LIN} \subseteq \{0, 1\}^{2k}$  is the predicate corresponding to the set of odd parity strings in  $\{0, 1\}^{2k}$ .

► **Theorem 1.2** (NP-hardness of Covering). *Let  $k \geq 2$ . Let  $P \subseteq 2k\text{-LIN}$  be any  $2k$ -bit predicate such there exists distributions  $\mathcal{P}_0, \mathcal{P}_1$  supported on  $\{0, 1\}^k$  with the following properties:*

1. *the marginals of  $\mathcal{P}_0$  and  $\mathcal{P}_1$  on all  $k$  coordinates is uniform,*
2. *every  $a \in \text{supp}(\mathcal{P}_0)$  has even parity and every  $b \in \text{supp}(\mathcal{P}_1)$  has odd parity and furthermore, both  $a \cdot b, b \cdot a \in P$ .*

*Then, unless  $\text{NP} \subseteq \text{DTIME}(2^{\text{poly} \log n})$ , for all  $\varepsilon \in (0, 1/2]$ , COVERING- $P$ -CSP( $2, \Omega(\log \log n)$ ) is not solvable in polynomial time.*

*Furthermore, the YES and NO instances of COVERING- $P$ -CSP( $2, \Omega(\log \log n)$ ) satisfy the following properties.*

- *YES Case: There are 2 assignments such that each of them covers  $1 - \varepsilon$  fraction of the constraints and they together cover the instance.*
- *NO Case: Even the  $2k\text{-LIN}$ -CSP instance with the same constraint graph as the given instance is not  $\Omega(\log \log n)$ -coverable.*

The furthermore clause in the soundness guarantee is in fact a strengthening for the following reason: if two predicates  $P, Q$  satisfy  $P \subseteq Q$  and  $\Phi$  is a  $c$ -coverable  $P$ -CSP instance, then the  $Q$ -CSP instance  $\Phi_{P \rightarrow Q}$  obtained by taking the constraint graph of  $\Phi$  and replacing each  $P$  constraint with the weaker  $Q$  constraint, is also  $c$ -coverable.

The following is a simple corollary of the above theorem.

► **Corollary 1.3.** *Let  $k \geq 2$  be even,  $x, y \in \{0, 1\}^k$  be distinct strings having even and odd parity respectively and  $\bar{x}, \bar{y}$  denote the complements of  $x$  and  $y$  respectively. For any predicate  $P$  satisfying*

$$2k\text{-LIN} \supseteq P \supseteq \{x \cdot y, x \cdot \bar{y}, \bar{x} \cdot y, \bar{x} \cdot \bar{y}, y \cdot x, y \cdot \bar{x}, \bar{y} \cdot x, \bar{y} \cdot \bar{x}\},$$

*unless  $\text{NP} \subseteq \text{DTIME}(2^{\text{poly} \log n})$ , the problem COVERING- $P$ -CSP( $2, \Omega(\log \log n)$ ) is not solvable in polynomial time.*

This corollary implies the covering hardness of 4-LIN predicate proved by Dinur and Kol [7] by setting  $x := 00$  and  $y := 01$ . With respect to the covering hardness of 4-LIN, we note that we can considerably simplify the proof of Dinur and Kol and in fact obtain an even stronger soundness guarantee (see Theorem below). The stronger soundness guarantee in the theorem below states that there are no large ( $\geq 1/\text{poly log } n$  fractional sized) independent sets in the constraint graph and hence, even the 4-NAE-CSP instance<sup>2</sup> with the same constraint graph as the given instance is not coverable using  $\Omega(\log \log n)$  assignments. Both the Dinur-Kol result and the above corollary only guarantee (in the soundness case) that the 4-LIN-CSP instance is not coverable.

► **Theorem 1.4 (Hardness of Covering 4-LIN).** *Assuming that  $NP \not\subseteq DTIME(2^{\text{poly log } n})$ , for all  $\varepsilon \in (0, 1)$ , there does not exist a polynomial time algorithm that can distinguish between 4-LIN-CSP instances of the following two types:*

- *YES Case: There are 2 assignments such that each of them covers  $1 - \varepsilon$  fraction of the constraints, and they together cover the entire instance.*
- *NO Case: The largest independent set in the constraint graph of the instance is of fractional size at most  $1/\text{poly log } n$ .*

## 1.2 Techniques

As one would expect, our proofs are very much inspired from the corresponding proofs in Dinur and Kol [7]. One of the main complications in the proof of Dinur and Kol [7] (as also in the earlier work of Guruswami, Håstad and Sudan [9]) was the one of handling several assignments simultaneously while proving the soundness analysis. For this purpose, both these works considered the rejection probability that all the assignments violated the constraint. This resulted in a very tedious expression for the rejection probability, which made the rest of the proof fairly involved. Khot [12] observed that this can be considerably simplified if one instead proved a stronger soundness guarantee that the largest independent set in the constraint graph is small (this might not always be doable, but in the cases when it is, it simplifies the analysis). We list below the further improvements in the proof that yield our Theorems 1.1, 1.2 and 1.4.

**Covering hardness of 4-LIN (Theorem 1.4):** The simplified proof of the covering hardness of 4-LIN follows directly from the above observation of using an independent set analysis instead of working with several assignments. In fact, this alternate proof eliminates the need for using results about correlated spaces [14], which was crucial in the Dinur-Kol setting. We further note that the quantitative improvement in the covering hardness ( $\Omega(\log \log n)$  over  $\Omega(\log \log \log n)$ ) comes from using a LABEL-COVER instance with a better smoothness property (see Theorem 2.5).

**Covering UG-hardness for non-odd predicates (Theorem 1.1):** Having observed that it suffices to prove an independent set analysis, we observed that only very mild conditions on the predicate are required to prove covering hardness. In particular, while Dinur and Kol used the Austrin-Mossel test [3] which required pairwise independence, we are able to import the long-code test of Bansal and Khot [4] which requires only 1-wise independence. We remark that the Bansal-Khot Test was designed for a specific predicate (hardness of finding independent sets in almost  $k$ -partite  $k$ -uniform hypergraphs) and had imperfect completeness.

<sup>2</sup> The  $k$ -NAE predicate over  $k$  bits is given by  $k\text{-NAE} = \{0, 1\}^k \setminus \{\bar{0}, \bar{1}\}$ .

Our improvement comes from observing that their test requires only 1-wise independence and furthermore that their completeness condition, though imperfect, can be adapted to give a 2-cover composed of 2 nearly satisfying assignments. This enlarges the class of non-odd predicates for which one can prove covering hardness (see Theorem 3.1). We then perform a sequence of reductions from this class of CSP instances to CSP instances over all non-odd predicates to obtain the final result. Interestingly, one of the open problems mentioned in the work of Dinur and Kol [7] was to devise “direct” reductions between covering problems. The reductions we employ, strictly speaking, are not “direct” reductions between covering problems, since they rely on a stronger soundness guarantee for the source instance (namely, large covering number even for the NAE instance on the same constraint graph), which we are able to prove in Theorem 3.1.

**Quasi-NP-hardness result (Theorem 1.2):** In this setting, we unfortunately are not able to use the simplification arising from using the independent set analysis and have to deal with the issue of several assignments. One of the steps in the 4-LIN proof of Dinur and Kol (as in several others results in this area) involves showing that an expression of the form  $\mathbb{E}_{(X,Y)} [F(X)F(Y)]$  is not too negative where  $(X, Y)$  is not necessarily a product distribution but the marginals on the  $X$  and  $Y$  parts are identical. Observe that if  $(X, Y)$  was a product distribution, then the above expression reduces to  $(\mathbb{E}_X [F(X)])^2$ , a positive quantity. Thus, the steps in the proof involve constructing a tailor-made distribution  $(X, Y)$  such that the error in going from the correlated probability space  $(X, Y)$  to the product distribution  $(X \otimes Y)$  is not too much. More precisely, the quantity

$$\left| \mathbb{E}_{(X,Y)} [F(X)F(Y)] - \mathbb{E}_X [F(X)] \mathbb{E}_Y [F(Y)] \right|,$$

is small. Dinur and Kol used a distribution tailor-made for the 4-LIN predicate and used an invariance principle for correlated spaces to bound the error while transforming it to a product distribution. Our improvement comes from observing that one could use an alternate invariance principle (see Theorem 2.8) that works with milder restrictions and hence works for a wider class of predicates. This invariance principle for correlated spaces (Theorem 2.8) is an adaptation of invariance principles proved by Wenner [17] and Guruswami and Lee [10] in similar contexts. The rest of the proof is similar to the 4-LIN covering hardness proof of Dinur and Kol.

### 1.3 Recent work on approximate coloring

We remark that recently, with the discovery of the short code [5], there has been a sequence of works [6, 8, 13, 16] which have considerably improved the status of the approximate coloring question, stated in the beginning of the introduction. In particular, we know that it is quasi-NP-hard to color a 2-colorable 8-uniform hypergraph with  $2^{(\log n)^c}$  colors for some constant  $c \in (0, 1)$ . Stated in terms of covering number, this result states that it is quasi-NP-hard to cover a 1-coverable 8-NAE-CSP instance with  $(\log n)^c$  assignments. It is to be noted that these results pertain to the covering complexity of specific predicates (such as NAE) whereas our results are concerned with classifying which predicates are hard to cover. It would be interesting if Theorem 1.2 and Theorem 1.4 can be improved to obtain similar hardness results (i.e.,  $\text{poly log } n$  as opposed to  $\text{poly log log } n$ ). The main bottleneck here seems to be reducing the uniformity parameter (namely, from 8).

## 1.4 Organization

The rest of the paper is organized as follows. We start with some preliminaries of LABEL-COVER, covering CSPs and Fourier analysis in § 2. Theorems 1.1, 1.2 and 1.4 are proved in Sections 3, 4 and 5 respectively.

## 2 Preliminaries

### 2.1 Covering CSPs

We will denote the set  $\{0, 1, \dots, q-1\}$  by  $[q]$ . For  $a \in [q]$ ,  $\bar{a} \in [q]^k$  is the element with  $a$  in all the  $k$  coordinates (where  $k$  and  $q$  will be implicit from the context).

► **Definition 2.1** (*P-CSP*). For a predicate  $P \subseteq [q]^k$ , an instance of *P-CSP* is given by a (hyper)graph  $G = (V, E)$ , referred to as the *constraint graph*, and a literals function  $L : E \rightarrow [q]^k$ , where  $V$  is a set of variables and  $E \subseteq V^k$  is a set of constraints. An assignment  $f : V \rightarrow [q]$  is said to *cover* a constraint  $e = (v_1, \dots, v_k) \in E$ , if  $(f(v_1), \dots, f(v_k)) + L(e) \in P$ , where addition is coordinate-wise modulo  $q$ . A set of assignments  $F = \{f_1, \dots, f_c\}$  is said to *cover*  $(G, L)$ , if for every  $e \in E$ , there is some  $f_i \in F$  that covers  $e$  and  $F$  is said to be a *c-covering* for  $G$ .  $G$  is said to be *c-coverable* if there is a *c-covering* for  $G$ . If  $L$  is not specified then it is the constant function which maps  $E$  to  $\bar{0}$ .

► **Definition 2.2** (*COVERING-P-CSP(c, s)*). For  $P \subseteq [q]^k$  and  $c, s \in \mathbb{N}$ , the *COVERING-P-CSP(c, s)* problem is, given a *c-coverable* instance  $(G = (V, E), L)$  of *P-CSP*, find an *s-covering*.

► **Definition 2.3** (*Odd*). A predicate  $P \subseteq [q]^k$  is *odd* if  $\forall x \in [q]^k, \exists a \in [q], x + \bar{a} \in P$ , where addition is coordinate-wise modulo  $q$ .

For odd predicates the covering problem is *trivially solvable*, since any CSP instance on such a predicate is *q-coverable* by the  $q$  translates of any assignment, i.e.,  $\{x + \bar{a} \mid a \in [q]\}$  is a *q-covering* for any assignment  $x \in [q]^k$ .

### 2.2 Label Cover

► **Definition 2.4** (*LABEL-COVER*). An instance  $G = (U, V, E, L, R, \{\pi_e\}_{e \in E})$  of the *LABEL-COVER* constraint satisfaction problem consists of a bi-regular bipartite graph  $(U, V, E)$ , two sets of alphabets  $L$  and  $R$  and a projection map  $\pi_e : R \rightarrow L$  for every edge  $e \in E$ . Given a labeling  $\ell : U \rightarrow L, \ell : V \rightarrow R$ , an edge  $e = (u, v)$  is said to be satisfied by  $\ell$  if  $\pi_e(\ell(v)) = \ell(u)$ .

$G$  is said to be *at most  $\delta$ -satisfiable* if every labeling satisfies at most a  $\delta$  fraction of the edges.  $G$  is said to be *c-coverable* if there exist  $c$  labelings such that for every vertex  $u \in U$ , one of the labelings satisfies all the edges incident on  $u$ .

An instance of *UNIQUE-GAMES* is a label cover instance where  $L = R$  and the constraints  $\pi$  are permutations.

The hardness of *LABEL-COVER* stated below follows from the PCP Theorem [2, 1], Raz's Parallel Repetition Theorem [15] and a structural property proved by Håstad [11, Lemma 6.9].

► **Theorem 2.5** (*Hardness of LABEL-COVER*). For every  $r \in \mathbb{N}$ , there is a deterministic  $n^{O(r)}$ -time reduction from a 3-SAT instance of size  $n$  to an instance  $G = (U, V, E, [L], [R], \{\pi_e\}_{e \in E})$  of *LABEL-COVER* with the following properties:

1.  $|U|, |V| \leq n^{O(r)}$ ;  $L, R \leq 2^{O(r)}$ ;  $G$  is bi-regular with degrees bounded by  $2^{O(r)}$ .



2. There exists a constant  $c_0 \in (0, 1/3)$  such that for any  $v \in V$  and  $\alpha \subseteq [R]$ , for a random neighbor  $u$ ,

$$\mathbb{E}_u [|\pi_{uv}(\alpha)|^{-1}] \leq |\alpha|^{-2c_0}.$$

This implies that

$$\forall v, \alpha, \quad Pr_u [|\pi_{uv}(\alpha)| < |\alpha|^{c_0}] \leq \frac{1}{|\alpha|^{c_0}}.$$

3. There is a constant  $d_0 \in (0, 1)$  such that,
- YES Case: If the 3-SAT instance is satisfiable, then  $G$  is 1-coverable.
  - NO Case: If the 3-SAT instance is unsatisfiable, then  $G$  is at most  $2^{-d_0 r}$ -satisfiable.

Our characterization of hardness of covering CSPs is based on the following conjecture due to Dinur and Kol [7].

► **Conjecture 2.6** (COVERING-UGC( $c$ )). *There exists  $c \in \mathbb{N}$  such that for every sufficiently small  $\delta > 0$  there exists  $L \in \mathbb{N}$  such that the following holds. Given an instance  $G = (U, V, E, [L], [L], \{\pi_e\}_{e \in E})$  of UNIQUE-GAMES it is NP-hard to distinguish between the following two cases:*

- YES Case: *There exist  $c$  assignments such that for every vertex  $u \in U$ , at least one of the assignments satisfies all the edges touching  $u$ .*
- NO Case: *Every assignment satisfies at most  $\delta$  fraction of the edge constraints.*

### 2.3 Analysis of Boolean Function over Probability Spaces

For a function  $f : \{0, 1\}^L \rightarrow \mathbb{R}$ , the Fourier decomposition of  $f$  is given by

$$f(x) = \sum_{\alpha \in \{0, 1\}^L} \hat{f}(\alpha) \chi_\alpha(x) \text{ where } \chi_\alpha(x) := (-1)^{\sum_{i=1}^L \alpha_i x_i} \text{ and } \hat{f}(\alpha) := \mathbb{E}_{x \in \{0, 1\}^L} f(x) \chi_\alpha(x).$$

We will use  $\alpha$ , also to denote the subset of  $[L]$  for which it is the characteristic vector. The Efron-Stein decomposition is a generalization of the Fourier decomposition to product distributions of arbitrary probability spaces. Let  $(\Omega, \mu)$  be a probability space and  $(\Omega^L, \mu^{\otimes L})$  be the corresponding product space. For a function  $f : \Omega^L \rightarrow \mathbb{R}$ , the Efron-Stein decomposition of  $f$  with respect to the product space is given by

$$f(x_1, \dots, x_L) = \sum_{\beta \subseteq [L]} f_\beta(x),$$

where  $f_\beta$  depends only on  $x_i$  for  $i \in \beta$  and for all  $\beta' \not\supseteq \beta$ ,  $a \in \Omega^{\beta'}$ ,  $\mathbb{E}_{x \in \mu^{\otimes R}} [f_\beta(x) \mid x_{\beta'} = a] = 0$ . We will be dealing with functions of the form  $f : \{0, 1\}^{dL} \rightarrow \mathbb{R}$  for  $d \in \mathbb{N}$  and  $d$ -to-1 functions  $\pi : [dL] \rightarrow [L]$ . We will also think of such functions as  $f : \prod_{i \in [L]} \Omega_i \rightarrow \mathbb{R}$  where  $\Omega_i = \{0, 1\}^d$  consists of the  $d$  coordinates  $j$  such that  $\pi(j) = i$ . An Efron-Stein decomposition of  $f : \prod_{i \in [L]} \Omega_i \rightarrow \mathbb{R}$  over the uniform distribution over  $\{0, 1\}^{dL}$ , can be obtained from the Fourier decomposition as

$$f_\beta(x) = \sum_{\alpha \subseteq [dL]: \pi(\alpha) = \beta} \hat{f}(\alpha) \chi_\alpha. \tag{2.1}$$

Let  $\|f\|_2 := \mathbb{E}_{x \in \mu^{\otimes L}} [f(x)^2]^{1/2}$  and  $\|f\|_\infty := \max_{x \in \Omega^{\otimes L}} |f(x)|$ . For  $i \in [L]$ , the influence of the  $i$ th coordinate on  $f$  is defined as follows.

$$\text{Inf}_i[f] := \mathbb{E}_{x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_L} \text{Var}_{x_i} [f(x_1, \dots, x_L)] = \sum_{\beta: i \in \beta} \|f_\beta\|_2^2.$$



For an integer  $d$ , the degree  $d$  influence is defined as

$$\text{Inf}_i^{\leq d}[f] := \sum_{\beta: i \in \beta, |\beta| \leq d} \|f_\beta\|_2^2.$$

It is easy to see that for Boolean functions, the sum of all the degree  $d$  influences is at most  $d$ .

Let  $(\Omega^k, \mu)$  be a probability space. Let  $S = \{x \in \Omega^k \mid \mu(x) > 0\}$ . We say that  $S \subseteq \Omega^k$  is *connected* if for every  $x, y \in S$ , there is a sequence of strings starting with  $x$  and ending with  $y$  such that every element in the sequence is in  $S$  and every two adjacent elements differ in exactly one coordinate.

► **Theorem 2.7** ([14, Proposition 6.4]). *Let  $(\Omega^k, \mu)$  be a probability space such that the support of the distribution  $\text{supp}(\mu) \subseteq \Omega^k$  is connected and the minimum probability of every atom in  $\text{supp}(\mu)$  is at least  $\alpha$  for some  $\alpha \in (0, \frac{1}{2}]$ . Then there exists continuous functions  $\bar{\Gamma} : (0, 1) \rightarrow (0, 1)$  and  $\underline{\Gamma} : (0, 1) \rightarrow (0, 1)$  such that the following holds: For every  $\varepsilon > 0$ , there exists  $\tau > 0$  and an integer  $d$  such that if a function  $f : \Omega^L \rightarrow [0, 1]$  satisfies*

$$\forall i \in [n], \text{Inf}_i^{\leq d}(f) \leq \tau$$

then

$$\underline{\Gamma}\left(\mathbb{E}_\mu[f]\right) - \varepsilon \leq \mathbb{E}_{(x_1, \dots, x_k) \sim \mu} \left[ \prod_{j=1}^k f(x_j) \right] \leq \bar{\Gamma}\left(\mathbb{E}_\mu[f]\right) + \varepsilon.$$

There exists an absolute constant  $C$  such that one can take  $\tau = \varepsilon^{C \frac{\log(1/\alpha) \log(1/\varepsilon)}{\varepsilon \alpha^2}}$  and  $d = \log(1/\tau) \log(1/\alpha)$ .

The following invariance principle for correlated spaces proved in Appendix A is an adaptation of similar invariance principles (c.f., [17, Theorem 3.12], [10, Lemma A.1]) to our setting.

► **Theorem 2.8** (Invariance Principle for correlated spaces). *Let  $(\Omega_1^k \times \Omega_2^k, \mu)$  be a correlated probability space such that the marginal of  $\mu$  on any pair of coordinates one each from  $\Omega_1$  and  $\Omega_2$  is a product distribution. Let  $\mu_1, \mu_2$  be the marginals of  $\mu$  on  $\Omega_1^k$  and  $\Omega_2^k$  respectively. Let  $X, Y$  be two random  $k \times L$  dimensional matrices chosen as follows: independently for every  $i \in [L]$ , the pair of columns  $(x^i, y^i) \in \Omega_1^k \times \Omega_2^k$  is chosen from  $\mu$ . Let  $x_i, y_i$  denote the  $i$ th rows of  $X$  and  $Y$  respectively. If  $F : \Omega_1^L \rightarrow [-1, +1]$  and  $G : \Omega_2^L \rightarrow [-1, +1]$  are functions such that*

$$\tau := \sqrt{\sum_{i \in [L]} \text{Inf}_i[F] \cdot \text{Inf}_i[G]} \quad \text{and} \quad \Gamma := \max \left\{ \sqrt{\sum_{i \in [L]} \text{Inf}_i[F]}, \sqrt{\sum_{i \in [L]} \text{Inf}_i[G]} \right\},$$

then

$$\left| \mathbb{E}_{(X, Y) \in \mu^{\otimes L}} \left[ \prod_{i \in [k]} F(x_i) G(y_i) \right] - \mathbb{E}_{X \in \mu_1^{\otimes L}} \left[ \prod_{i \in [k]} F(x_i) \right] \mathbb{E}_{Y \in \mu_2^{\otimes L}} \left[ \prod_{i \in [k]} G(y_i) \right] \right| \leq 2^{O(k)} \Gamma \tau. \quad (2.2)$$

### 3 UG Hardness of Covering

In this section, we prove the following theorem, which in turn implies Theorem 1.1 (see below for proof).

► **Theorem 3.1.** *Let  $[q]$  be any constant sized alphabet and  $k \geq 2$ . Recall that  $\text{NAE} := [q]^k \setminus \{\bar{b} \mid b \in [q]\}$ . Let  $P \subseteq [q]^k$  be a predicate such that there exists  $a \in \text{NAE}$  and  $\text{NAE} \supset P \supseteq \{a + \bar{b} \mid b \in [q]\}$ . Assuming  $\text{COVERING-UGC}(c)$ , for every sufficiently small constant  $\delta > 0$  it is NP-hard to distinguish between  $P$ -CSP instances  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$  of the following two cases:*

- *YES Case:  $\mathcal{G}$  is  $2c$ -coverable.*
- *NO Case:  $\mathcal{G}$  does not have an independent set of fractional size  $\delta$ .*

**Proof of Theorem 1.1.** Let  $Q$  be an arbitrary non odd predicate, i.e,  $Q \subseteq [q]^k \setminus \{h + \bar{b} \mid b \in [q]\}$  for some  $h \in [q]^k$ . Consider the predicate  $Q' \subseteq [q]^k$  defined as  $Q' := Q - h$ . Observe that  $Q' \subseteq \text{NAE}$ . Given any  $Q'$ -CSP instance  $\Phi$  with literals function  $L(e) = \bar{0}$ , consider the  $Q$ -CSP instance  $\Phi_{Q' \rightarrow Q}$  with literals function  $M$  given by  $M(e) := \bar{h}, \forall e$ . It has the same constraint graph as  $\Phi$ . Clearly,  $\Phi$  is  $c$ -coverable iff  $\Phi_{Q' \rightarrow Q}$  is  $c$ -coverable. Thus, it suffices to prove the result for any predicate  $Q' \subseteq \text{NAE}$  with literals function  $L(e) = \bar{0}^3$ . We will consider two cases, both of which will follow from Theorem 3.1.

Suppose the predicate  $Q'$  satisfies  $Q' \supseteq \{a + \bar{b} \mid b \in [q]\}$  for some  $a \in [q]^k$ . Then this predicate  $Q'$  satisfies the hypothesis of Theorem 3.1 and the theorem follows if we show that the soundness guarantee of Theorem 3.1 implies that in Theorem 1.1. Any instance in the NO case of Theorem 3.1, is not  $t := \log_q(1/\delta)$ -coverable even on the  $\text{NAE}$ -CSP instance with the same constraint graph. This is because any  $t$ -covering for the  $\text{NAE}$ -CSP instance gives a coloring of the constraint graph using  $q^t$  colors, by choosing the color of every variable to be a string of length  $t$  and having the corresponding assignments in each position in  $[t]$ . Hence the  $Q'$ -CSP instance is also not  $t$ -coverable.

Suppose  $Q' \not\supseteq \{a + \bar{b} \mid b \in [q]\}$  for all  $a \in [q]^k$ . Then consider the predicate  $P = \{a + \bar{b} \mid a \in Q', b \in [q]\} \subseteq \text{NAE}$ . Notice that  $P$  satisfies the conditions of Theorem 3.1 and if the  $P$ -CSP instance is  $t$ -coverable then the  $Q'$ -CSP instance is  $qt$ -coverable. Hence an YES instance of Theorem 3.1 maps to a  $2cq$ -coverable  $Q$ -CSP instance and NO instance maps to an instance with covering number at least  $\log_q(1/\delta)$ . ◀

We now prove Theorem 3.1 by giving a reduction from an instance  $G = (U, V, E, [L], [L], \{\pi_e\}_{e \in E})$  of  $\text{UNIQUE-GAMES}$  as in Definition 2.4, to an instance  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$  of a  $P$ -CSP for any predicate  $P$  that satisfies the conditions mentioned. As stated in the introduction, we adapt the long-code test of Bansal and Khot [4] for proving the hardness of finding independent sets in almost  $k$ -partite  $k$ -uniform hypergraphs to our setting. The set of variables  $\mathcal{V}$  is  $V \times [q]^{2L}$ . Any assignment to  $\mathcal{V}$  is given by a set of functions  $f_v : [q]^{2L} \rightarrow [q]$ , for each  $v \in V$ . The set of constraints  $\mathcal{E}$  is given by the following test which checks whether  $f_v$ 's are long codes of a good labeling to  $V$ . There is a constraint corresponding to all the variables that are queried together by the test.

### Long Code Test $\mathcal{T}_1$

1. Choose  $u \in U$  uniformly and  $k$  neighbors  $w_1, \dots, w_k \in V$  of  $u$  uniformly and independently at random.
2. Choose a random matrix  $X$  of dimension  $k \times 2L$  as follows. Let  $X^i$  denote the  $i^{\text{th}}$  column of  $X$ . Independently for each  $i \in [L]$ , choose  $(X^i, X^{i+L})$  uniformly at random from the

<sup>3</sup> This observation [7] that the cover- $Q$  problem for any non-odd predicate  $Q$  is equivalent to the cover- $Q'$  problem where  $Q' \subseteq \text{NAE}$  shows the centrality of the  $\text{NAE}$  predicate in understanding the covering complexity of any non-odd predicate.

set

$$S := \{(y, y') \in [q]^k \times [q]^k \mid y \in \{a + \bar{b} \mid b \in [q]\} \vee y' \in \{a + \bar{b} \mid b \in [q]\}\}. \quad (3.1)$$

3. Let  $x_1, \dots, x_k$  be the rows of matrix  $X$ . Accept iff

$$(f_{w_1}(x_1 \circ \pi_{uw_1}), f_{w_2}(x_2 \circ \pi_{uw_2}), \dots, f_{w_k}(x_k \circ \pi_{uw_k})) \in P,$$

where  $x \circ \pi$  is the string defined as  $(x \circ \pi)(i) := x_{\pi(i)}$  for  $i \in [L]$  and  $(x \circ \pi)(i) := x_{\pi(i-L)+L}$  otherwise.

► **Lemma 3.2** (Completeness). *If the UNIQUE-GAMES instance  $G$  is  $c$ -coverable then the  $P$ -CSP instance  $\mathcal{G}$  is  $2c$ -coverable.*

**Proof.** Let  $\ell_1, \dots, \ell_c : U \cup V \rightarrow [L]$  be a  $c$ -covering for  $G$  as described in Definition 2.4. We will show that the  $2c$  assignments given by  $f_v^i(x) := x_{\ell_i(v)}$ ,  $g_v^i(x) := x_{\ell_i(v)+L}$ ,  $i = 1, \dots, c$  form a  $2c$ -covering of  $\mathcal{G}$ . Consider any  $u \in U$  and let  $\ell_i$  be the labeling that covers all the edges incident on  $u$ . For any  $(u, w_j)_{j \in \{1, \dots, k\}} \in E$  and  $X$  chosen by the long code test  $\mathcal{T}_1$ , the vector  $(f_{w_1}^i(x_1 \circ \pi_{uw_1}), \dots, f_{w_k}^i(x_k \circ \pi_{uw_k}))$  gives the  $\ell_i(u)$ th column of  $X$ . Similarly the above expression corresponding to  $g^i$  gives the  $(\ell_i(u) + L)$ th column of the matrix  $X$ . Since, for all  $i \in [L]$ , either  $i$ th column or  $(i+L)$ th column of  $X$  contains element from  $\{a + \bar{b} \mid b \in [q]\} \subseteq P$ , either  $(f_{w_1}^i(x_1 \circ \pi_{uw_1}), \dots, f_{w_k}^i(x_k \circ \pi_{uw_k})) \in P$  or  $(g_{w_1}^i(x_1 \circ \pi_{uw_1}), \dots, g_{w_k}^i(x_k \circ \pi_{uw_k})) \in P$ . Hence the set of  $2c$  assignments  $\{f_v^i, g_v^i\}_{i \in \{1, \dots, c\}}$  covers all constraints in  $\mathcal{G}$ . ◀

To prove soundness, we show that the set  $S$ , as defined in Equation (3.1), is connected, so that Theorem 2.7 is applicable. For this, we view  $S \subseteq [q]^k \times [q]^k$  as a subset of  $([q]^2)^k$  as follows: the element  $(y, y') \in S$  is mapped to the element  $((y_1, y'_1), \dots, (y_k, y'_k)) \in ([q]^2)^k$ .

► **Claim 3.3.** *Let  $\Omega = [q]^2$ . The set  $S \subset \Omega^k$  is connected.*

**Proof.** Consider any  $x := (x^1, x^2), y := (y^1, y^2) \in S \subset [q]^k \times [q]^k$ . Suppose both  $x^1, y^1 \in \{a + \bar{b} \mid b \in [q]\}$ , then it is easy to come up with a sequence of strings belonging to  $S$ , starting with  $x$  and ending with  $y$  such that consecutive strings differ in at most 1 coordinate. Now suppose  $x^1, y^2 \in \{a + \bar{b} \mid b \in [q]\}$ . First we come up with a sequence from  $x$  to  $z := (z^1, z^2)$  such that  $z^1 := x^1$  and  $z^2 = y^2$ , and then another sequence for  $z$  to  $y$ . ◀

► **Lemma 3.4** (Soundness). *For every constant  $\delta > 0$ , there exists a constant  $s$  such that, if  $G$  is at most  $s$ -satisfiable then  $\mathcal{G}$  does not have an independent set of size  $\delta$ .*

**Proof.** Let  $I \subseteq \mathcal{V}$  be an independent set of fractional size  $\delta$  in the constraint graph. For every variable  $v \in V$ , let  $f_v : [q]^{2L} \rightarrow \{0, 1\}$  be the indicator function of the independent set restricted to the vertices that correspond to  $v$ . For a vertex  $u \in U$ , let  $N(u) \subseteq V$  be the set of neighbors of  $u$  and define  $f_u(x) := \mathbb{E}_{w \in N(u)} [f_w(x \circ \pi_{uw})]$ . Since  $I$  is an independent set, we have

$$0 = \mathbb{E}_{u, w_1, \dots, w_k} \mathbb{E}_{X \sim \mathcal{T}_1} \left[ \prod_{i=1}^k f_{w_i}(x_i \circ \pi_{uw_i}) \right] = \mathbb{E}_u \mathbb{E}_{X \sim \mathcal{T}_1} \left[ \prod_{i=1}^k f_u(x_i) \right]. \quad (3.2)$$

Since the bipartite graph  $(U, V, E)$  is left regular and  $|I| \geq \delta|V|$ , we have  $\mathbb{E}_{u, x} [f_u(x)] \geq \delta$ . By an averaging argument, for at least  $\frac{\delta}{2}$  fraction of the vertices  $u \in U$ ,  $\mathbb{E}_x [f_u(x)] \geq \frac{\delta}{2}$ . Call a vertex  $u \in U$  *good* if it satisfies this property. A string  $x \in [q]^{2L}$  can be thought as an element from  $([q]^2)^L$  by grouping the pair of coordinates  $x_i, x_{i+L}$ . Let  $\bar{x} \in ([q]^2)^L$  denotes this grouping of  $x$ , i.e.,  $j$ th coordinate of  $\bar{x}$  is  $(x_j, x_{j+L}) \in [q]^2$ . With this grouping, the

function  $f_u$  can be viewed as  $f_u : ([q]^2)^L \rightarrow \{0, 1\}$ . From Equation (3.2), we have that for any  $u \in U$ ,

$$\mathbb{E}_{X \sim \mathcal{T}_1} \left[ \prod_{i=1}^k f_u(\bar{x}_i) \right] = 0.$$

By Claim 3.3, for all  $j \in [L]$  the tuple  $((\bar{x}_1)_j, \dots, (\bar{x}_k)_j)$  (corresponding to columns  $(X^j, X^{j+L})$  of  $X$ ) is sampled from a distribution whose support is a connected set. Hence for a good vertex  $u \in U$ , we can apply Theorem 2.7 with  $\varepsilon = \underline{\Gamma}(\delta/2)/2$  to get that there exists  $j \in [L], d \in \mathbb{N}, \tau > 0$  such that  $\text{Inf}_j^{\leq d}(f_u) > \tau$ . We will use this fact to give a randomized labeling for  $G$ . Labels for vertices  $w \in V, u \in U$  will be chosen uniformly and independently from the sets

$$\text{Lab}(w) := \left\{ i \in [L] \mid \text{Inf}_i^{\leq d}(f_w) \geq \frac{\tau}{2} \right\}, \text{Lab}(u) := \left\{ i \in [L] \mid \text{Inf}_i^{\leq d}(f_u) \geq \tau \right\}.$$

By the above argument (using Theorem 2.7), we have that for a good vertex  $u$ ,  $\text{Lab}(u) \neq \emptyset$ . Furthermore, since the sum of degree  $d$  influences is at most  $d$ , the above sets have size at most  $2d/\tau$ . Now, for any  $j \in \text{Lab}(u)$ , we have

$$\begin{aligned} \tau < \text{Inf}_j^{\leq d}[f_u] &= \sum_{S: j \in S, |S| \leq d} \|f_{u,S}\|^2 = \sum_{S: j \in S, |S| \leq d} \left\| \mathbb{E}_{w \in N(u)} [f_{w, \pi_{uw}^{-1}(S)}] \right\|^2 \quad (\text{By Definition.}) \\ &\leq \sum_{S: j \in S, |S| \leq d} \mathbb{E}_{w \in N(u)} \left\| f_{w, \pi_{uw}^{-1}(S)} \right\|^2 = \mathbb{E}_{w \in N(u)} \text{Inf}_{\pi_{uw}^{-1}(j)}^{\leq d}[f_w]. \quad (\text{By Convexity of square.}) \end{aligned}$$

Hence, by another averaging argument, there exists at least  $\frac{\tau}{2}$  fraction of neighbors  $w$  of  $u$  such that  $\text{Inf}_{\pi_{uw}^{-1}(j)}^{\leq d}(f_w) \geq \frac{\tau}{2}$  and hence  $\pi_{uw}^{-1}(j) \in \text{Lab}(w)$ . Therefore, for a good vertex  $u \in U$ , at least  $\frac{\tau}{2} \frac{\tau}{2d}$  fraction of edges incident on  $u$  are satisfied in expectation. Also, at least  $\frac{\delta}{2}$  fraction of vertices in  $U$  are good, it follows that the expected fraction of edges that are satisfied by this random labeling is at least  $\frac{\delta}{2} \frac{\tau}{2} \frac{\tau}{2d}$ . Choosing  $s < \frac{\delta}{2} \frac{\tau}{2} \frac{\tau}{2d}$  completes the proof.  $\blacktriangleleft$

## 4 NP-Hardness of Covering

In this section, we prove Theorem 1.2. We give a reduction from an instance of a LABEL-COVER,  $G = (U, V, E, [L], [R], \{\pi_e\}_{e \in E})$  as in Definition 2.4, to a  $P$ -CSP instance  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$  for any predicate  $P$  that satisfies the conditions mentioned in Theorem 1.2. The reduction and proof is similar to that of Dinur and Kol [7]. The main difference is that they used a test and invariance principle very specific to the 4-LIN predicate, while we show that a similar analysis can be performed under milder conditions on the test distribution.

We assume that  $R = dL$  and  $\forall i \in [L], e \in E, |\pi_e^{-1}(i)| = d$ . This is done just for simplifying the notation and the proof does not depend upon it. The set of variables  $\mathcal{V}$  is  $V \times \{0, 1\}^{2R}$ . Any assignment to  $\mathcal{V}$  is given by a set of functions  $f_v : \{0, 1\}^{2R} \rightarrow \{0, 1\}$ , for each  $v \in V$ . The set of constraints  $\mathcal{E}$  is given by the following test which checks whether  $f_v$ 's are long codes of a good labeling to  $V$ .

### Long Code Test $\mathcal{T}_2$

1. Choose  $u \in U$  uniformly and  $v, w \in V$  neighbors of  $u$  uniformly and independently at random. For  $i \in [L]$ , let  $B_{uv}(i) := \pi_{uv}^{-1}(i), B'_{uv}(i) := R + \pi_{uv}^{-1}(i)$  and similarly for  $w$ .
2. Choose matrices  $X, Y$  of dimension  $k \times 2dL$  as follows. For  $S \subseteq [2dL]$ , we denote by  $X|_S$  the submatrix of  $X$  restricted to the columns  $S$ . Independently for each  $i \in [L]$ , choose  $c_1 \in \{0, 1\}$  uniformly and

- a. if  $c_1 = 0$ , choose  $(X|_{B_{uv}(i) \cup B'_{uv}(i)}, Y|_{B_{uw}(i) \cup B'_{uw}(i)})$  from  $\mathcal{P}_0^{\otimes 2d} \otimes \mathcal{P}_1^{\otimes 2d}$ ,
  - b. if  $c_1 = 1$ , choose  $(X|_{B_{uv}(i) \cup B'_{uv}(i)}, Y|_{B_{uw}(i) \cup B'_{uw}(i)})$  from  $\mathcal{P}_1^{\otimes 2d} \otimes \mathcal{P}_0^{\otimes 2d}$ .
3. Perturb  $X, Y$  as follows. Independently for each  $i \in [L]$ , choose  $c_2 \in \{*, 0, 1\}$  as follows:  $\Pr[c_2 = *] = 1 - 2\varepsilon$ , and  $\Pr[c_2 = 1] = \Pr[c_2 = 0] = \varepsilon$ . Perturb the  $i$ th matrix block  $(X|_{B_{uv}(i) \cup B'_{uv}(i)}, Y|_{B_{uw}(i) \cup B'_{uw}(i)})$  as follows:
    - a. if  $c_2 = *$ , leave the matrix block  $(X|_{B_{uv}(i) \cup B'_{uv}(i)}, Y|_{B_{uw}(i) \cup B'_{uw}(i)})$  unperturbed,
    - b. if  $c_2 = 0$ , choose  $(X|_{B'_{uv}(i)}, Y|_{B'_{uw}(i)})$  uniformly from  $\{0, 1\}^{k \times d} \times \{0, 1\}^{k \times d}$ ,
    - c. if  $c_2 = 1$ , choose  $(X|_{B_{uv}(i)}, Y|_{B_{uw}(i)})$  uniformly from  $\{0, 1\}^{k \times d} \times \{0, 1\}^{k \times d}$ .
  4. Let  $x_1, \dots, x_k$  and  $y_1, \dots, y_k$  be the rows of the matrices  $X$  and  $Y$  respectively. Accept if

$$(f_v(x_1), \dots, f_v(x_k), f_w(y_1), \dots, f_w(y_k)) \in P.$$

► **Lemma 4.1** (Completeness). *If  $G$  is an YES instance of LABEL-COVER, then there exists  $f, g$  such that each of them covers  $1 - \varepsilon$  fraction of  $\mathcal{E}$  and they together cover all of  $\mathcal{E}$ .*

**Proof.** Let  $\ell : U \cup V \rightarrow [L] \cup [R]$  be a labeling to  $G$  that satisfies all the constraints. Consider the assignments  $f_v(x) := x_{\ell(v)}$  and  $g_w(x) := x_{R+\ell(v)}$  for each  $v \in V$ . First consider the assignment  $f$ . For any  $(u, v), (u, w) \in E$  and  $x_1, \dots, x_k, y_1, \dots, y_k$  chosen by the long code test  $\mathcal{T}_2$ ,  $(f_v(x_1), \dots, f_v(x_k), f_w(y_1), \dots, f_w(y_k))$  gives the  $\ell(v)$ th and  $\ell(w)$ th column of the matrices  $X$  and  $Y$  respectively. Since  $\pi_{uv}(\ell(v)) = \pi_{uw}(\ell(w))$ , they are jointly distributed either according to  $\mathcal{P}_0 \otimes \mathcal{P}_1$  or  $\mathcal{P}_1 \otimes \mathcal{P}_0$  after Step 2. The probability that these rows are perturbed in Step 3c is at most  $\varepsilon$ . Hence with probability  $1 - \varepsilon$  over the test distribution,  $f$  is accepted. A similar argument shows that the test accepts  $g$  with probability  $1 - \varepsilon$ . Note that in Step 3, the columns given by  $f, g$ , are never re-sampled uniformly together. Hence they together cover  $\mathcal{G}$ . ◀

Now we will show that if  $G$  is a NO instance of LABEL-COVER then no  $t$  assignments can cover the  $2k$ -LIN-CSP with constraint hypergraph  $\mathcal{G}$ . For the rest of the analysis, we will use  $+1, -1$  instead of the symbols  $0, 1$ . Suppose for contradiction, there exist  $t$  assignments  $f_1, \dots, f_t : \{\pm 1\}^{2R} \rightarrow \{\pm 1\}$  that form a  $t$ -cover to  $\mathcal{G}$ . The probability that all the  $t$  assignments are rejected in Step 4 is

$$\mathbb{E}_{u,v,w} \mathbb{E}_{\mathcal{T}_2} \left[ \prod_{i=1}^t \frac{1}{2} \left( \prod_{j=1}^k f_{i,v}(x_j) f_{i,w}(y_j) + 1 \right) \right] = \frac{1}{2^t} + \frac{1}{2^t} \sum_{\emptyset \subset S \subseteq \{1, \dots, t\}} \mathbb{E}_{u,v,w} \mathbb{E}_{\mathcal{T}_2} \left[ \prod_{j=1}^k f_{S,v}(x_j) f_{S,w}(y_j) \right]. \quad (4.1)$$

where  $f_{S,v}(x) := \prod_{i \in S} f_{i,v}(x)$ . Since the  $t$  assignments form a  $t$ -cover, the LHS in Equation (4.1) is 0 and hence, there exists an  $S \neq \emptyset$  such that

$$\mathbb{E}_{u,v,w} \mathbb{E}_{\mathcal{T}_2} \left[ \prod_{j=1}^k f_{S,v}(x_j) f_{S,w}(y_j) \right] \leq -1/(2^t - 1). \quad (4.2)$$

The following lemma shows that this is not possible if  $t$  is not too large, thus proving that there does not exist a  $t$ -cover.

► **Lemma 4.2** (Soundness). *Let  $c_0 \in (0, 1)$  be the constant from Theorem 2.5 and  $S \subseteq \{1, \dots, t\}, |S| > 0$ . If  $G$  is at most  $s$ -satisfiable then*

$$\mathbb{E}_{u,v,w} \mathbb{E}_{X,Y \in \mathcal{T}_2} \left[ \prod_{i=1}^k f_{S,v}(x_i) f_{S,w}(y_i) \right] \geq -O(k s^{c_0/8}) - 2^{O(k)} \frac{s^{(1-3c_0)/8}}{\varepsilon^{3/2c_0}}.$$

**Proof.** Notice that for a fixed  $u$ , the distribution of  $X$  and  $Y$  have identical marginals. Hence the value of the above expectation, if calculated according to a distribution which is the direct product of the marginals, is positive. We will first show that the expectation can change by at most  $O(ks^{c_0/8})$  in moving to an *attenuated* version of the functions (see Claim 4.3). Then we will show that the error incurred by changing the distribution to the product distribution of the marginals has absolute value at most  $2^{O(k)} \frac{s^{(1-3c_0)/8}}{\varepsilon^{3/2c_0}}$  (see Claim 4.5). This is done by showing that there is a labeling to  $G$  that satisfies an  $s$  fraction of the constraints if the error is more than  $2^{O(k)} \frac{s^{(1-3c_0)/8}}{\varepsilon^{3/2c_0}}$ .

For the rest of the analysis, we write  $f_v$  and  $f_w$  instead of  $f_{S,v}$  and  $f_{S,w}$  respectively. Let  $f_v = \sum_{\alpha \subseteq [2R]} \widehat{f}_v(\alpha) \chi_\alpha$  be the Fourier decomposition of the function and for  $\gamma \in (0, 1)$ , let  $T_{1-\gamma} f_v := \sum_{\alpha \subseteq [2R]} (1-\gamma)^{|\alpha|} \widehat{f}_v(\alpha) \chi_\alpha$ . The following claim is similar to a lemma of Dinur and Kol [7, Lemma 4.11]. The only difference in the proof is that, we use the *smoothness* from Property 2 of Theorem 2.5 (which was shown by Håstad [11, Lemma 6.9]).

► **Claim 4.3.** Let  $\gamma := s^{(c_0+1)/4} \varepsilon^{1/c_0}$  where  $c_0$  is the constant from Theorem 2.5.

$$\left| \mathbb{E}_{u,v,w} \mathbb{E}_{\mathcal{T}_2} \left[ \prod_{i=1}^k f_v(x_i) f_w(y_i) \right] - \mathbb{E}_{u,v,w} \mathbb{E}_{\mathcal{T}_2} \left[ \prod_{i=1}^k T_{1-\gamma} f_v(x_i) T_{1-\gamma} f_w(y_i) \right] \right| \leq O(ks^{c_0/8}).$$

**Proof.** We will add the  $T_{1-\gamma}$  operator to one function at a time and upper bound the absolute value of the error incurred each time by  $O(s^{c_0/8})$ . The total error is at most  $2k$  times the error in adding  $T_{1-\gamma}$  to one function. Hence, it suffices to prove the following

$$\left| \mathbb{E}_{u,v,w} \mathbb{E}_{\mathcal{T}_2} \left[ \prod_{i=1}^k f_v(x_i) f_w(y_i) \right] - \mathbb{E}_{u,v,w} \mathbb{E}_{\mathcal{T}_2} \left[ \left( \prod_{i=1}^{k-1} f_v(x_i) f_w(y_i) \right) f_v(x_k) T_{1-\gamma} f_w(y_k) \right] \right| \leq O(s^{c_0/8}). \quad (4.3)$$

Recall that  $X, Y$  denote the matrices chosen by test  $\mathcal{T}_2$ . Let  $Y_{-k}$  be the matrix obtained from  $Y$  by removing the  $k$ th row and  $F_{u,v,w}(X, Y_{-k}) := \left( \prod_{i=1}^{k-1} f_v(x_i) f_w(y_i) \right) f_v(x_k)$ . Then, (4.3) can be rewritten as

$$\left| \mathbb{E}_{u,v,w} \mathbb{E}_{\mathcal{T}_2} [F_{u,v,w}(X, Y_{-k}) (I - T_{1-\gamma}) f_w(y_k)] \right| \leq O(s^{c_0/8}). \quad (4.4)$$

Let  $U$  be the operator that maps functions on the variable  $y_k$ , to one on the variables  $(X, Y_{-k})$  defined by

$$(Uf)(X, Y_{-k}) := \mathbb{E}_{y_k | X, Y_{-k}} f(y_k).$$

Let  $G_{u,v,w}(X, Y_{-k}) := (U(I - T_{1-\gamma})f_w)(X, Y_{-k})$ . Note that  $\mathbb{E}_{y \in \{0,1\}^{2R}} G_{u,v,w}(y) = 0$ . For the rest of the analysis, fix  $u, v, w$  chosen by the test. We will omit the subscript  $u, v, w$  from now on for notational convenience. The domain of  $G$  can be thought of as  $(\{0, 1\}^{2k-1})^{2dL}$  and the test distribution on any row is independent across the blocks  $\{B_{uv}(i) \cup B'_{uv}(i)\}_{i \in [L]}$ . We now think of  $G$  as having domain  $\prod_{i \in [L]} \Omega_i$  where  $\Omega_i = (\{0, 1\}^{2k-1})^{2d}$  corresponds to the set of rows in  $B_{uv}(i) \cup B'_{uv}(i)$ . Let the following be the Efron-Stein decomposition of  $G$  with respect to  $\mathcal{T}_2$ ,

$$G(X, Y_{-k}) = \sum_{\alpha \subseteq [L]} G_\alpha(X, Y_{-k}).$$

The following technical claim follows from a result similar to [7, Lemma 4.7] and then using [14, Proposition 2.12]. We defer its proof to Appendix B.

► **Claim 4.4.** For  $\alpha \subseteq [L]$

$$\|G_\alpha\|^2 \leq (1-\varepsilon)^{|\alpha|} \sum_{\beta \subseteq [2R]: \tilde{\pi}_{uw}(\beta)=\alpha} \left(1 - (1-\gamma)^{2|\beta|}\right) \widehat{f}_w(\beta)^2 \quad (4.5)$$

where  $\tilde{\pi}_{uw}(\beta) := \{i \in [L] : \exists j \in [R], (j \in \beta \vee j + R \in \beta) \wedge \pi_{uw}(j) = i\}$ .

Substituting the Efron-Stein decomposition of  $G, F$  into the LHS of (4.4) gives

$$\begin{aligned} \left| \mathbb{E}_{u,v,w} \mathbb{E}_{T_2} [F_{u,v,w}(X, Y_{-k}) (I - T_{1-\gamma}) f_w(y_k)] \right| &= \left| \mathbb{E}_{u,v,w} \mathbb{E}_{T_2} F(X, Y_{-k}) G(X, Y_{-k}) \right| \\ &\stackrel{\text{(By orthonormality of Efron-Stein decomposition)}}{=} \left| \mathbb{E}_{u,v,w} \sum_{\alpha \subseteq [L]} \mathbb{E}_{T_2} F_\alpha(X, Y_{-k}) G_\alpha(X, Y_{-k}) \right| \\ \text{(By Cauchy-Schwarz inequality)} &\leq \mathbb{E}_{u,v,w} \sqrt{\sum_{\alpha \subseteq [L]} \|F_\alpha\|^2} \cdot \sqrt{\sum_{\alpha \subseteq [L]} \|G_\alpha\|^2} \\ \text{(Using } \sum_{\alpha \subseteq [L]} \|F_\alpha\|^2 = \|F\|_2^2 = 1) &\leq \mathbb{E}_{u,v,w} \sqrt{\sum_{\alpha \subseteq [L]} \|G_\alpha\|^2}. \end{aligned}$$

Using concavity of square root and substituting for  $\|G_\alpha\|^2$  from Equation (4.5), we get that the above is upper bounded by

$$\sqrt{\sum_{\alpha \subseteq [L]} \sum_{\substack{\beta \subseteq [2R]: \\ \tilde{\pi}_{uw}(\beta)=\alpha}} \underbrace{\mathbb{E}_{u,v,w} (1-\varepsilon)^{|\alpha|} \left(1 - (1-\gamma)^{2|\beta|}\right) \widehat{f}_w(\beta)^2}_{=: \text{Term}_{u,w}(\alpha, \beta)}}.$$

We will now break the above summation into three different parts and bound each part separately.

$$\begin{aligned} \Theta_0 &:= \mathbb{E}_{u,w} \sum_{\alpha, \beta: |\alpha| \geq \frac{1}{\varepsilon s^{c_0/4}}} \text{Term}_{u,w}(\alpha, \beta), & \Theta_1 &:= \mathbb{E}_{u,w} \sum_{\substack{\alpha, \beta: |\alpha| < \frac{1}{\varepsilon s^{c_0/4}} \\ |\beta| \leq \frac{2}{s^{1/4} \varepsilon^{1/c_0}}} \text{Term}_{u,w}(\alpha, \beta), \\ \Theta_2 &:= \mathbb{E}_{u,w} \sum_{\substack{\alpha, \beta: |\alpha| < \frac{1}{\varepsilon s^{c_0/4}} \\ |\beta| > \frac{2}{s^{1/4} \varepsilon^{1/c_0}}} \text{Term}_{u,w}(\alpha, \beta). \end{aligned}$$

**Upper bounding  $\Theta_0$ :** When  $|\alpha| > \frac{1}{\varepsilon s^{c_0/4}}$ ,  $(1-\varepsilon)^{|\alpha|} < s^{c_0/4}$ . Also since  $f_w$  is  $\{+1, -1\}$  valued, sum of squares of Fourier coefficient is 1. Hence  $|\Theta_0| < s^{c_0/4}$ .

**Upper bounding  $\Theta_1$ :** When  $|\beta| \leq \frac{2}{s^{1/4} \varepsilon^{1/c_0}}$ ,

$$1 - (1-\gamma)^{2|\beta|} \leq 1 - \left(1 - \frac{4}{s^{1/4} \varepsilon^{1/c_0}} \gamma\right) = \frac{4}{s^{1/4} \varepsilon^{1/c_0}} \gamma = 4s^{c_0/4}.$$

Again since the sum of squares of Fourier coefficients is 1,  $|\Theta_1| \leq 4s^{c_0/4}$ .

**Upper bounding  $\Theta_2$ :** From Property 2 of Theorem 2.5, we have that for any  $v \in V$  and  $\beta$  with  $|\beta| > \frac{2}{s^{1/4} \varepsilon^{1/c_0}}$ , the probability that  $|\tilde{\pi}_{uv}(\beta)| < 1/\varepsilon s^{c_0/4}$ , for a random neighbor  $u$ , is at most  $\varepsilon s^{c_0/4}$ . Hence  $|\Theta_2| \leq s^{c_0/4}$ .

◀

Fix  $u, v, w$  chosen by the test. Recall that we thought of  $f_v$  as having domain  $\prod_{i \in [L]} \Omega_i$  where  $\Omega_i = \{0, 1\}^{2d}$  corresponds to the set of coordinates in  $B_{uv}(i) \cup B'_{uv}(i)$ . Since the grouping of coordinates depends on  $u$ , we define  $\overline{\text{Inf}}_i^u[f_v] := \text{Inf}_i[f_v]$  where  $i \in [L]$  for explicitness. From Equation (2.1),

$$\overline{\text{Inf}}_i^u[f_v] = \sum_{\alpha \subseteq [2dL]: i \in \tilde{\pi}_{uv}(\alpha)} \widehat{f}_v(\alpha)^2,$$

where  $\tilde{\pi}_{uv}(\alpha) := \{i \in [L] : \exists j \in [R], (j \in \alpha \vee j + R \in \alpha) \wedge \pi_{uv}(j) = i\}$ .

► **Claim 4.5.** Let  $\tau_{u,v,w} := \sum_{i \in [L]} \overline{\text{Inf}}_i^u[T_{1-\gamma}f_v] \cdot \overline{\text{Inf}}_i^u[T_{1-\gamma}f_w]$ .

$$\begin{aligned} \mathbb{E}_{u,v,w} \left| \mathbb{E}_{\mathcal{T}_2} \left[ \prod_{i=1}^k T_{1-\gamma}f_v(x_i)T_{1-\gamma}f_w(y_i) \right] - \mathbb{E}_{\mathcal{T}_2} \left[ \prod_{i=1}^k T_{1-\gamma}f_v(x_i) \right] \mathbb{E}_{\mathcal{T}_2} \left[ \prod_{i=1}^k T_{1-\gamma}f_w(y_i) \right] \right| \\ \leq 2^{O(k)} \sqrt{\frac{\mathbb{E}_{u,v,w} \tau_{u,v,w}}{\gamma}}. \end{aligned}$$

**Proof.** It is easy to check that  $\sum_{i \in [L]} \overline{\text{Inf}}_i^u[T_{1-\gamma}f_v] \leq 1/\gamma$  (c.f., [17, Lemma 1.13]). For any  $u, v, w$ , since the test distribution satisfies the conditions of Theorem 2.8, we get

$$\left| \mathbb{E}_{\mathcal{T}_2} \left[ \prod_{i=1}^k T_{1-\gamma}f_v(x_i)T_{1-\gamma}f_w(y_i) \right] - \mathbb{E}_{\mathcal{T}_2} \left[ \prod_{i=1}^k T_{1-\gamma}f_v(x_i) \right] \mathbb{E}_{\mathcal{T}_2} \left[ \prod_{i=1}^k T_{1-\gamma}f_w(y_i) \right] \right| \leq 2^{O(k)} \sqrt{\frac{\tau_{u,v,w}}{\gamma}}.$$

The claim follows by taking expectation over  $u, v, w$  and using the concavity of square root. ◀

From Claim 4.5 and Claim 4.3 and using the fact the the marginals of the test distribution  $\mathcal{T}_2$  on  $(x_1, \dots, x_k)$  is the same as marginals on  $(y_1, \dots, y_k)$ , for  $\gamma := s^{(c_0+1)/4} \varepsilon^{1/c_0}$ , we get

$$\mathbb{E}_{u,v,w} \mathbb{E}_{X,Y \in \mathcal{T}_2} \left[ \prod_{i=1}^k f_v(x_i) f_w(y_i) \right] \geq -O(ks^{c_0/8}) - 2^{O(k)} \sqrt{\frac{\mathbb{E}_{u,v,w} \tau_{u,v,w}}{\gamma}} + \mathbb{E}_u \left( \mathbb{E}_v \mathbb{E}_{\mathcal{T}_2} \left[ \prod_{i=1}^k T_{1-\gamma}f_v(x_i) \right] \right)^2. \quad (4.6)$$

If  $\tau_{u,v,w}$  in expectation is large, there is a standard way of decoding the assignments to a labeling to the label cover instance, as shown in Claim 4.6.

► **Claim 4.6.** If  $G$  is an at most  $s$ -satisfiable instance of LABEL-COVER then

$$\mathbb{E}_{u,v,w} \tau_{u,v,w} \leq \frac{s}{\gamma^2}.$$

**Proof.** Note that  $\sum_{\alpha \subseteq [2R]} (1-\gamma)^{|\alpha|} \widehat{f}_v(\alpha)^2 \leq 1$ . We will give a randomized labeling to the LABEL-COVER instance.

For each  $v \in V$ , choose a random  $\alpha \subseteq [2R]$  with probability  $(1-\gamma)^{|\alpha|} \widehat{f}_v(\alpha)^2$  and assign a uniformly random label  $j$  in  $\alpha$  to  $v$ ; if the label  $j \geq R$ , change the label to  $j - R$  and with the remaining probability assign an arbitrary label. For  $u \in U$ , choose a random neighbor  $w \in V$  and a random  $\beta \subseteq [2R]$  with probability  $(1-\gamma)^{|\beta|} \widehat{f}_w(\beta)^2$ , choose a random label  $\ell$  in  $\beta$  and assign the label  $\tilde{\pi}_{uw}(\ell)$  to  $u$ . With the remaining probability, assign an arbitrary label. The fraction of edges satisfied by this labeling is at least

$$\mathbb{E}_{u,v,w} \sum_{i \in [L]} \sum_{(\alpha, \beta): i \in \tilde{\pi}_{uv}(\alpha), i \in \tilde{\pi}_{uw}(\beta)} \frac{(1-\gamma)^{|\alpha|+|\beta|}}{|\alpha| \cdot |\beta|} \widehat{f}_v(\alpha)^2 \widehat{f}_w(\beta)^2.$$



Using the fact that  $1/r \geq \gamma(1-\gamma)^r$  for every  $r > 0$  and  $\gamma \in [0, 1]$ , we lower bound  $\frac{1}{|\alpha|}$  and  $\frac{1}{|\beta|}$  by  $\gamma(1-\gamma)^{|\alpha|}$  and  $\gamma(1-\gamma)^{|\beta|}$  respectively. The above is then lower bounded by

$$\gamma^2 \mathbb{E}_{u,v,w} \sum_{i \in [L]} \left( \sum_{\alpha: i \in \tilde{\pi}_{uv}(\alpha)} (1-\gamma)^{2|\alpha|} \widehat{f}_v(\alpha)^2 \right) \left( \sum_{\beta: i \in \tilde{\pi}_{uw}(\beta)} (1-\gamma)^{2|\beta|} \widehat{f}_w(\beta)^2 \right) = \gamma^2 \mathbb{E}_{u,v,w} \tau_{u,v,w}.$$

Since  $G$  is at most  $s$ -satisfiable, the labeling can satisfy at most  $s$  fraction of constraints and the above equation is upper bounded by  $s$ . ◀

Lemma 4.2 follows from the above claim and Equation 4.6. ◀

**Proof of Theorem 1.2.** Using Theorem 2.5, the size of the CSP instance  $\mathcal{G}$  produced by the reduction is  $N = n^r 2^{2^{O(r)}}$  and the parameter  $s \leq 2^{-d_0 r}$ . Setting  $r = \Theta(\log \log n)$ , gives that  $N = 2^{\text{poly}(\log n)}$  for a constant  $k$ . Lemma 4.2 and Equation 4.2 imply that

$$O(k s^{c_0/8}) + 2^{O(k)} \frac{s^{(1-3c_0)/8}}{\varepsilon^{3/2c_0}} \geq \frac{1}{2^t - 1}.$$

Since  $k$  is a constant, this gives that  $t = \Omega(\log \log n)$ . ◀

## 5 Improvement to covering hardness of 4-LIN

In this section, we prove Theorem 1.4. We give a reduction from an instance of LABEL-COVER,  $G = (U, V, E, [L], [R], \{\pi_e\}_{e \in E})$  as in Definition 2.4, to a 4-LIN-CSP instance  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ . The set of variables  $\mathcal{V}$  is  $V \times \{0, 1\}^{2R}$ . Any assignment to  $\mathcal{V}$  is given by a set of functions  $f_v : \{0, 1\}^{2R} \rightarrow \{0, 1\}$ , for each  $v \in V$ . The set of constraints  $\mathcal{E}$  is given by the following test which checks whether  $f_v$ 's are long codes of a good labeling to  $V$ .

### Long Code Test $\mathcal{T}_3$

1. Choose  $u \in U$  uniformly and neighbors  $v, w \in V$  of  $u$  uniformly and independently at random.
2. Choose  $x, x', z, z'$  uniformly and independently from  $\{0, 1\}^{2R}$  and  $y$  from  $\{0, 1\}^{2L}$ . Choose  $(\eta, \eta') \in \{0, 1\}^{2L} \times \{0, 1\}^{2L}$  as follows: Independently for each  $i \in [L]$ ,  $(\eta_i, \eta_{L+i}, \eta'_i, \eta'_{L+i})$  is set to
  - a.  $(0, 0, 0, 0)$  with probability  $1 - 2\varepsilon$ ,
  - b.  $(1, 0, 1, 0)$  with probability  $\varepsilon$  and
  - c.  $(0, 1, 0, 1)$  with probability  $\varepsilon$ .
3. For  $y \in \{0, 1\}^{2L}$ , let  $y \circ \pi_{uv} \in \{0, 1\}^{2R}$  be the string such that  $(y \circ \pi_{uv})_i := y_{\pi_{uv}(i)}$  for  $i \in [R]$  and  $(y \circ \pi_{uv})_i := y_{\pi_{uv}(i-R)+L}$  otherwise. Given  $\eta \in \{0, 1\}^{2L}, z \in \{0, 1\}^{2R}$ , the string  $\eta \circ \pi_{uv} \cdot z \in \{0, 1\}^{2R}$  is obtained by taking coordinate-wise product of  $\eta \circ \pi_{uv}$  and  $z$ . Accept iff

$$f_v(x) + f_v(x + y \circ \pi_{uv} + \eta \circ \pi_{uv} \cdot z) + f_w(x') + f_w(x' + y \circ \pi_{uw} + \eta' \circ \pi_{uw} \cdot z' + 1) = 1 \pmod{2}. \quad (5.1)$$

(Here by addition of strings, we mean the coordinate-wise sum modulo 2.)

► **Lemma 5.1 (Completeness).** *If  $G$  is an YES instance of LABEL-COVER, then there exists  $f, g$  such that each of them covers  $1 - \varepsilon$  fraction of  $\mathcal{E}$  and they together cover all of  $\mathcal{E}$ .*

**Proof.** Let  $\ell : U \cup V \rightarrow [L] \cup [R]$  be a labeling to  $G$  that satisfies all the constraints. Consider the assignments given by  $f_v(x) := x_{\ell(v)}$  and  $g_v(x) := x_{R+\ell(v)}$  for each  $v \in V$ . On input  $f_v$ , for any pair of edges  $(u, v), (u, w) \in E$ , and  $x, x', z, z', \eta, \eta', y$  chosen by the long code test  $\mathcal{T}_3$ , the LHS in (5.1) evaluates to

$$x_{\ell(v)} + x_{\ell(v)} + y_{\ell(u)} + \eta_{\ell(u)} z_{\ell(v)} + x'_{\ell(w)} + x'_{\ell(w)} + y_{\ell(u)} + \eta'_{\ell(u)} z'_{\ell(w)} + 1 = \eta_{\ell(u)} z_{\ell(v)} + \eta'_{\ell(u)} z'_{\ell(w)} + 1.$$

Similarly for  $g_v$ , the expression evaluates to  $\eta_{L+\ell(u)} z_{R+\ell(v)} + \eta'_{L+\ell(u)} z'_{R+\ell(w)} + 1$ . Since  $(\eta_i, \eta'_i) = (0, 0)$  with probability  $1 - \varepsilon$ , each of  $f, g$  covers  $1 - \varepsilon$  fraction of  $\mathcal{E}$ . Also for  $i \in [L]$  whenever  $(\eta_i, \eta'_i) = (1, 1)$ ,  $(\eta_{L+i}, \eta'_{L+i}) = (0, 0)$  and vice versa. So one of the two evaluations above is  $1 \pmod{2}$ . Hence the pair of assignment  $f, g$  cover  $\mathcal{E}$ .  $\blacktriangleleft$

**► Lemma 5.2 (Soundness).** *Let  $c_0$  be the constant from Theorem 2.5. If  $G$  is at most  $s$ -satisfiable with  $s < \frac{\delta^{10/c_0+5}}{4}$ , then any independent set in  $\mathcal{G}$  has fractional size at most  $\delta$ .*

**Proof.** Let  $I \subseteq \mathcal{V}$  be an independent set of fractional size  $\delta$  in the constraint graph  $\mathcal{G}$ . For every variable  $v \in V$ , let  $f_v : \{0, 1\}^{2R} \rightarrow \{0, 1\}$  be the indicator function of the independent set restricted to the vertices that correspond to  $v$ . Since  $I$  is an independent set, we have

$$\mathbb{E}_{\substack{u, v, w \\ x, x', \\ z, z', \\ \eta, \eta', y}} [f_v(x) f_v(x + y \circ \pi_{uv} + \eta \circ \pi_{uv} \cdot z) f_w(x') f_w(x' + y \circ \pi_{uw} + \eta' \circ \pi_{uw} \cdot z') + 1] = 0. \quad (5.2)$$

For  $\alpha \subseteq [2R]$ , let  $\pi_{uv}^{\oplus}(\alpha) \subseteq [2L]$  be the set containing elements  $i \in [2L]$  such that if  $i < L$  there are an odd number of  $j \in [R] \cap \alpha$  with  $\pi_{uv}(j) = i$  and if  $i \geq L$  there are an odd number of  $j \in ([2R] \setminus [R]) \cap \alpha$  with  $\pi_{uv}(j - R) = i - L$ . It is easy to see that  $\chi_{\alpha}(y \circ \pi_{uv}) = \chi_{\pi_{uv}^{\oplus}(\alpha)}(y)$ . Expanding  $f_v$  in the Fourier basis and taking expectation over  $x, x'$  and  $y$ , we get that

$$\mathbb{E}_{u, v, w} \sum_{\alpha, \beta \subseteq [2R]: \pi_{uv}^{\oplus}(\alpha) = \pi_{uw}^{\oplus}(\beta)} \widehat{f}_v(\alpha)^2 \widehat{f}_w(\beta)^2 (-1)^{|\beta|} \mathbb{E}_{z, z', \eta, \eta'} [\chi_{\alpha}(\eta \circ \pi_{uv} \cdot z) \chi_{\beta}(\eta' \circ \pi_{uw} \cdot z')] = 0. \quad (5.3)$$

Now the expectation over  $z, z'$  simplifies as

$$\mathbb{E}_{u, v, w} \sum_{\alpha, \beta \subseteq [2R]: \pi_{uv}^{\oplus}(\alpha) = \pi_{uw}^{\oplus}(\beta)} \widehat{f}_v(\alpha)^2 \widehat{f}_w(\beta)^2 (-1)^{|\beta|} \underbrace{\Pr_{\eta, \eta'} [\alpha \cdot (\eta \circ \pi_{uv}) = \beta \cdot (\eta' \circ \pi_{uw}) = \bar{0}]}_{=: \text{Term}_{u, v, w}(\alpha, \beta)} = 0, \quad (5.4)$$

where we think of  $\alpha, \beta$  as the characteristic vectors in  $\{0, 1\}^{2R}$  of the corresponding sets. We will now break up the above summation into different parts and bound each part separately. For a projection  $\pi : [R] \rightarrow [L]$ , define  $\tilde{\pi}(\alpha) := \{i \in [L] : \exists j \in [R], (j \in \alpha \vee j + R \in \alpha) \wedge (\pi(j) =$

$i)$ . We need the following definitions.

$$\begin{aligned}\Theta_0 &:= \mathbb{E}_{u,v,w} \sum_{\substack{\alpha,\beta: \\ \pi_{uv}^\oplus(\alpha) = \pi_{uw}^\oplus(\beta) = \emptyset}} \text{Term}_{u,v,w}(\alpha, \beta), \\ \Theta_1 &:= \mathbb{E}_{u,v,w} \sum_{\substack{\alpha,\beta: \\ \pi_{uv}^\oplus(\alpha) = \pi_{uw}^\oplus(\beta) \neq \emptyset, \\ \max\{|\alpha|, |\beta|\} \leq 2/\delta^{5/c_0}}} \text{Term}_{u,v,w}(\alpha, \beta), \\ \Theta_2 &:= \mathbb{E}_{u,v,w} \sum_{\substack{\alpha,\beta: \\ \pi_{uv}^\oplus(\alpha) = \pi_{uw}^\oplus(\beta) \neq \emptyset, \\ \max\{|\pi_{uv}(\alpha)|, |\pi_{uw}(\beta)|\} \geq 1/\delta^5}} \text{Term}_{u,v,w}(\alpha, \beta), \\ \Theta_3 &:= \mathbb{E}_{u,v,w} \sum_{\substack{\alpha,\beta: \\ \pi_{uv}^\oplus(\alpha) = \pi_{uw}^\oplus(\beta) \neq \emptyset, \\ \max\{|\alpha|, |\beta|\} > 2/\delta^{5/c_0}, \\ \max\{|\tilde{\pi}_{uv}(\alpha)|, |\tilde{\pi}_{uw}(\beta)|\} < 1/\delta^5}} \text{Term}_{u,v,w}(\alpha, \beta).\end{aligned}$$

**Lower bounding  $\Theta_0$ :** If  $\pi_{uw}^\oplus(\beta) = \emptyset$ , then  $|\beta|$  is even. Hence, all the terms in  $\Theta_0$  are positive and

$$\Theta_0 \geq \mathbb{E}_{u,v,w} \text{Term}_{u,v,w}(0, 0) = \mathbb{E}_u \left( \mathbb{E}_v \widehat{f}_v(0)^2 \right)^2 \geq \left( \mathbb{E}_{u,v} \widehat{f}_v(0) \right)^4 = \delta^4.$$

**Upper bounding  $\Theta_1$ :** Consider the following strategy for labeling vertices  $u \in U$  and  $v \in V$ . For  $u \in U$ , pick a random neighbor  $v$ , choose  $\alpha$  with probability  $\widehat{f}_v(\alpha)^2$  and set its label to a random element in  $\tilde{\pi}_{uv}(\alpha)$ . For  $w \in V$ , choose  $\beta$  with probability  $\widehat{f}_w(\beta)^2$  and set its label to a random element of  $\beta$ . If the label  $j \geq R$ , change the label to  $j - R$ . The probability that a random edge  $(u, w)$  of the label cover is satisfied by this labeling is

$$\begin{aligned}\mathbb{E}_{u,v,w} \sum_{\substack{\alpha,\beta: \\ \tilde{\pi}_{uv}(\alpha) \cap \tilde{\pi}_{uw}(\beta) \neq \emptyset}} \widehat{f}_v(\alpha)^2 \widehat{f}_w(\beta)^2 \frac{1}{|\tilde{\pi}_{uv}(\alpha)| \cdot |\beta|} &\geq \mathbb{E}_{u,v,w} \sum_{\substack{\alpha,\beta: \\ \pi_{uv}^\oplus(\alpha) = \pi_{uw}^\oplus(\beta) \neq \emptyset \\ \max\{|\alpha|, |\beta|\} \leq 2/\delta^{5/c_0}}} \widehat{f}_v(\alpha)^2 \widehat{f}_w(\beta)^2 \frac{\delta^{10/c_0}}{4} \\ &\geq |\Theta_1| \cdot \frac{\delta^{10/c_0}}{4}.\end{aligned}$$

Since the instance is at most  $s$ -satisfiable, the above is upper bounded by  $s$ . Choosing  $s < \frac{\delta^{10/c_0+5}}{4}$ , will imply  $|\Theta_1| \leq \delta^5$ .

**Upper bounding  $\Theta_2$ :** Suppose  $|\tilde{\pi}_{uv}(\alpha)| \geq 1/\delta^5$ , then note that

$$\Pr_{\eta, \eta'}[\alpha \cdot (\eta \circ \pi_{uv}) = \beta \cdot (\eta' \circ \pi_{uw}) = 0] \leq \Pr[\alpha \cdot (\eta \circ \pi_{uv}) = 0] \leq (1 - \varepsilon)^{|\tilde{\pi}_{uv}(\alpha)|} \leq (1 - \varepsilon)^{1/\delta^5}.$$

Since the sum of squares of Fourier coefficients of  $f$  is less than 1 and  $\varepsilon$  is a constant, we get that  $|\Theta_2| \leq 1/2^{\Omega(1/\delta^5)} < O(\delta^5)$ .

**Upper bounding  $\Theta_3$ :** From the third property of Theorem 2.5, we have that for any  $v \in V$  and  $\alpha \subseteq [2R]$  with  $|\alpha| > 2/\delta^{5/c_0}$ , the probability that  $|\tilde{\pi}_{uv}(\alpha)| < 1/\delta^5$ , for a random neighbor  $u$  of  $v$ , is at most  $\delta^5$ . Hence  $|\Theta_3| \leq \delta^5$ .

On substituting the above bounds in Equation (5.4), we get that  $\delta^4 - O(\delta^5) \leq 0$  which gives a contradiction for small enough  $\delta$ . Hence there is no independent set in  $\mathcal{G}$  of size  $\delta$ . ◀

**Proof of Theorem 1.4.** From Theorem 2.5, the size of the CSP instance  $\mathcal{G}$  produced by the reduction is  $N = n^r 2^{2^{O(r)}}$  and the parameter  $s \leq 2^{-d_0 r}$ . Setting  $r = \Theta(\log \log n)$ , gives that  $N = 2^{\text{poly}(\log n)}$  and the size of the largest independent set  $\delta = 1/\text{poly}(\log n) = 1/\text{poly}(\log N)$ . ◀

---

## References

- 1 Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *J. ACM*, 45(3):501–555, May 1998. (Preliminary version in *33rd FOCS*, 1992).
- 2 Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: A new characterization of NP. *J. ACM*, 45(1):70–122, January 1998. (Preliminary version in *33rd FOCS*, 1992).
- 3 Per Austrin and Elchanan Mossel. Approximation resistant predicates from pairwise independence. *Comput. Complexity*, 18(2):249–271, 2009. (Preliminary version in *23rd IEEE Conference on Computational Complexity*, 2008).
- 4 Nikhil Bansal and Subhash Khot. Inapproximability of hypergraph vertex cover and applications to scheduling problems. In Samson Abramsky, Cyril Gavoille, Claude Kirchner, Friedhelm Meyer auf der Heide, and Paul G. Spirakis, editors, *Proc. 37th International Colloq. of Automata, Languages and Programming (ICALP), Part I*, volume 6198 of *LNCS*, pages 250–261. Springer, 2010.
- 5 Boaz Barak, Parikshit Gopalan, Johan Håstad, Raghu Meka, Prasad Raghavendra, and David Steurer. Making the long code shorter. In *Proc. 53rd IEEE Symp. on Foundations of Comp. Science (FOCS)*, pages 370–379, 2012.
- 6 Irit Dinur and Venkatesan Guruswami. PCPs via low-degree long code and hardness for constrained hypergraph coloring. In *Proc. 54th IEEE Symp. on Foundations of Comp. Science (FOCS)*, pages 340–349, 2013.
- 7 Irit Dinur and Gillat Kol. Covering CSPs. In *Proc. 28th IEEE Conf. on Computational Complexity*, pages 207–218, 2013.
- 8 Venkat Guruswami, Prahladh Harsha, Johan Håstad, Srikanth Srinivasan, and Girish Varma. Super-polylogarithmic hypergraph coloring hardness via low-degree long codes. In *Proc. 46th ACM Symp. on Theory of Computing (STOC)*, pages 614–623, 2014.
- 9 Venkatesan Guruswami, Johan Håstad, and Madhu Sudan. Hardness of approximate hypergraph coloring. *SIAM J. Computing*, 31(6):1663–1686, 2002. (Preliminary Version in *41st FOCS*, 2000).
- 10 Venkatesan Guruswami and Euiwoong Lee. Strong inapproximability results on balanced rainbow-colorable hypergraphs. In *Proc. 26th Annual ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 822–836, 2015.
- 11 Johan Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, July 2001. (Preliminary version in *29th STOC*, 1997).
- 12 Subhash Khot. Hardness results for coloring 3-colorable 3-uniform hypergraphs. In *Proc. 43rd IEEE Symp. on Foundations of Comp. Science (FOCS)*, pages 23–32, 2002.
- 13 Subhash Khot and Rishi Saket. Hardness of coloring 2-colorable 12-uniform hypergraphs with  $2^{(\log n)^{\Omega(1)}}$  colors. In *Proc. 55th IEEE Symp. on Foundations of Comp. Science (FOCS)*, pages 206–215, 2014.
- 14 Elchanan Mossel. Gaussian bounds for noise correlation of functions. *Geom. Funct. Anal.*, 19(6):1713–1756, 2010. (Preliminary version in *49th FOCS*, 2008).
- 15 Ran Raz. A parallel repetition theorem. *SIAM J. Computing*, 27(3):763–803, June 1998. (Preliminary version in *27th STOC*, 1995).
- 16 Girish Varma. A note on reducing uniformity in Khot-Saket hypergraph coloring hardness reductions. (manuscript), 2014.

- 17 Cenny Wenner. Circumventing  $d$ -to-1 for approximation resistance of satisfiable predicates strictly containing parity of width at least four. *Theory Comput.*, 9(23):703–757, 2013. (Preliminary version in *APPROX*, 2012).

**A Invariance Principle for correlated spaces**

**Theorem 2.8 (Invariance Principle for correlated spaces) [Restated].** *Let  $(\Omega_1^k \times \Omega_2^k, \mu)$  be a correlated probability space such that the marginal of  $\mu$  on any pair of coordinates one each from  $\Omega_1$  and  $\Omega_2$  is a product distribution. Let  $\mu_1, \mu_2$  be the marginals of  $\mu$  on  $\Omega_1^k$  and  $\Omega_2^k$  respectively. Let  $X, Y$  be two random  $k \times L$  dimensional matrices chosen as follows: independently for every  $i \in [L]$ , the pair of columns  $(x^i, y^i) \in \Omega_1^k \times \Omega_2^k$  is chosen from  $\mu$ . Let  $x_i, y_i$  denote the  $i$ th rows of  $X$  and  $Y$  respectively. If  $F : \Omega_1^L \rightarrow [-1, +1]$  and  $G : \Omega_2^L \rightarrow [-1, +1]$  are functions such that*

$$\tau := \sqrt{\sum_{i \in [L]} \text{Inf}_i[F] \cdot \text{Inf}_i[G]} \quad \text{and} \quad \Gamma := \max \left\{ \sqrt{\sum_{i \in [L]} \text{Inf}_i[F]}, \sqrt{\sum_{i \in [L]} \text{Inf}_i[G]} \right\},$$

then

$$\left| \mathbb{E}_{(X,Y) \in \mu^{\otimes L}} \left[ \prod_{i \in [k]} F(x_i) G(y_i) \right] - \mathbb{E}_{X \in \mu_1^{\otimes L}} \left[ \prod_{i \in [k]} F(x_i) \right] \mathbb{E}_{Y \in \mu_2^{\otimes L}} \left[ \prod_{i \in [k]} G(y_i) \right] \right| \leq 2^{O(k)} \Gamma \tau. \quad (\text{A.1})$$

**Proof.** We will prove the theorem by using the hybrid argument. For  $i \in [L+1]$ , let  $X^{(i)}, Y^{(i)}$  be distributed according to  $(\mu_1 \otimes \mu_2)^{\otimes i} \otimes \mu^{\otimes L-i}$ . Thus,  $(X^{(0)}, Y^{(0)}) = (X, Y)$  is distributed according to  $\mu^{\otimes L}$  while  $(X^{(L)}, Y^{(L)})$  is distributed according to  $(\mu_1 \otimes \mu_2)^{\otimes L}$ . For  $i \in [L]$ , define

$$\text{err}_i := \left| \mathbb{E}_{X^{(i)}, Y^{(i)}} \left[ \prod_{j=1}^k F(x_j^{(i)}) G(y_j^{(i)}) \right] - \mathbb{E}_{X^{(i+1)}, Y^{(i+1)}} \left[ \prod_{j=1}^k F(x_j^{(i+1)}) G(y_j^{(i+1)}) \right] \right|. \quad (\text{A.2})$$

The left hand side of Equation (2.2) is upper bounded by  $\sum_{i \in [L]} \text{err}_i$ . Now for a fixed  $i$ , we will bound  $\text{err}_i$ . We use the Efron-Stein decomposition of  $F, G$  to split them into two parts: the part which depends on the  $i$ th input and the part independent of the  $i$ th input.

$$F = F_0 + F_1 \quad \text{where} \quad F_0 := \sum_{\alpha: i \notin \alpha} F_\alpha \quad \text{and} \quad F_1 := \sum_{\alpha: i \in \alpha} F_\alpha.$$

$$G = G_0 + G_1 \quad \text{where} \quad G_0 := \sum_{\beta: i \notin \beta} G_\beta \quad \text{and} \quad G_1 := \sum_{\beta: i \in \beta} G_\beta.$$

Note that  $\text{Inf}_i[F] = \|F_1\|_2^2$  and  $\text{Inf}_i[G] = \|G_1\|_2^2$ . Furthermore, the functions  $F_0$  and  $F_1$  are bounded since  $F_0(x) = \mathbb{E}_{x'} [F(x')] | x'_{[L] \setminus i} = x_{[L] \setminus i} \in [-1, +1]$  and  $F_1(x) = F(x) - F_0(x) \in [-2, +2]$ . For  $a \in \{0, 1\}^k$ , let  $F_a(X) := \prod_{j=1}^k F_{a_j}(x_j)$ . Similarly  $G_0, G_1$  are bounded and  $G_a$  defined analogously. Substituting these definitions in Equation (A.2) and expanding the products gives

$$\text{err}_i = \left| \sum_{a,b \in \{0,1\}^k} \left( \mathbb{E}_{X^{(i)}, Y^{(i)}} \left[ F_a(X^{(i)}) G_b(Y^{(i)}) \right] - \mathbb{E}_{X^{(i+1)}, Y^{(i+1)}} \left[ F_a(X^{(i+1)}) G_b(Y^{(i+1)}) \right] \right) \right|.$$

Since both the distributions are identical on  $(\Omega_1^k)^{\otimes L}$  and  $(\Omega_2^k)^{\otimes L}$ , all terms with  $a = \bar{0}$  or  $b = \bar{0}$  are zero. Because  $\mu$  is uniform on any pair of coordinates on each from the  $\Omega_1$  and  $\Omega_2$  sides, terms with  $|a| = |b| = 1$  also evaluates to zero. Now consider the remaining terms with  $|a|, |b| \geq 1, |a| + |b| > 2$ . Consider one such term where  $a_1, a_2 = 1$  and  $b_1 = 1$ . In this case, by Cauchy-Schwarz inequality we have that

$$\left| \mathbb{E}_{X^{(i-1)}, Y^{(i-1)}} \left[ F_a(X^{(i-1)}) G_b(Y^{(i-1)}) \right] \right| \leq \sqrt{\mathbb{E} F_1(x_1)^2 G_1(y_1)^2} \cdot \|F_1\|_2 \cdot \left\| \prod_{j>2} F_{a_j} \right\|_{\infty} \cdot \left\| \prod_{j>1} G_{b_j} \right\|_{\infty}.$$

From the facts that the marginal of  $\mu$  to any pair of coordinates one each from  $\Omega_1$  and  $\Omega_2$  sides are uniform,  $\text{Inf}_i[F] = \|F_1\|_2^2$  and  $|F_0(x)|, |F_1(x)|, |G_0(x)|, |G_1(x)|$  are all bounded by 2, the right side of above becomes

$$\sqrt{\mathbb{E} F_1(x_1)^2} \sqrt{\mathbb{E} G_1(y_1)^2} \cdot \|F_1\|_2 \cdot \left\| \prod_{j>2} F_{a_j} \right\|_{\infty} \cdot \left\| \prod_{j>1} G_{b_j} \right\|_{\infty} \leq \sqrt{\text{Inf}_i[F]^2 \text{Inf}_i[G]} \cdot 2^{2k}.$$

All the other terms corresponding to other  $(a, b)$  which are at most  $2^{2k}$  in number, are bounded analogously. Hence,

$$\begin{aligned} \sum_{i \in [L]} \text{err}_i &\leq 2^{4k} \sum_{i \in [L]} \left( \sqrt{\text{Inf}_i[F]^2 \text{Inf}_i[G]} + \sqrt{\text{Inf}_i[F] \text{Inf}_i[G]^2} \right) \\ &= 2^{4k} \sum_{i \in [L]} \sqrt{\text{Inf}_i[F] \text{Inf}_i[G]} \left( \sqrt{\text{Inf}_i[F]} + \sqrt{\text{Inf}_i[G]} \right). \end{aligned}$$

By applying the Cauchy-Schwarz inequality, followed by a triangle inequality, we obtain

$$\sum_{i \in [L]} \text{err}_i \leq 2^{4k} \sqrt{\sum_{i \in [L]} \text{Inf}_i[F] \text{Inf}_i[G]} \left( \sqrt{\sum_{i \in [L]} \text{Inf}_i[F]} + \sqrt{\sum_{i \in [L]} \text{Inf}_i[G]} \right).$$

Thus, proved.  $\blacktriangleleft$

## B Proof of Claim 4.4

We will be reusing the notation introduced in the long code test  $\mathcal{T}_2$ . We denote the  $k \times 2d$  dimensional matrix  $X|_{B(i) \cup B'(i)}$  by  $X^i$  and  $Y|_{B(i) \cup B'(i)}$  by  $Y^i$ . Also by  $X_j^i$ , we mean the  $j$ th row of the matrix  $X^i$  and  $Y_{-k}^i$  is the first  $k-1$  rows of  $Y^i$ . The spaces of the random variables  $X^i, X_j^i, Y_{-k}^i$  will be denoted by  $\mathcal{X}^i, \mathcal{X}_j^i, \mathcal{Y}_{-k}^i$ .

Before we proceed to the proof of claim, we need a few definitions and lemmas related to correlated spaces defined by Mossel [14].

► **Definition B.1.** Let  $(\Omega_1 \times \Omega_2, \mu)$  be a finite correlated space, the correlation between  $\Omega_1$  and  $\Omega_2$  with respect to  $\mu$  is defined as

$$\rho(\Omega_1, \Omega_2; \mu) := \max_{\substack{f: \Omega_1 \rightarrow \mathbb{R}, \mathbb{E}[f]=0, \mathbb{E}[f^2] \leq 1 \\ g: \Omega_2 \rightarrow \mathbb{R}, \mathbb{E}[g]=0, \mathbb{E}[g^2] \leq 1}} \mathbb{E} [|f(x)g(y)|].$$

► **Definition B.2 (Markov Operator).** Let  $(\Omega_1 \times \Omega_2, \mu)$  be a finite correlated space, the Markov operator, associated with this space, denoted by  $U$ , maps a function  $g: \Omega_2 \rightarrow \mathbb{R}$  to functions  $Ug: \Omega_1 \rightarrow \mathbb{R}$  by the following map:

$$(Ug)(x) := \mathbb{E}_{(X,Y) \sim \mu} [g(Y) \mid X = x].$$

The following results (from [14]) provide a way to upper bound correlation of a correlated spaces.

► **Lemma B.3** ([14, Lemma 2.8]). *Let  $(\Omega_1 \times \Omega_2, \mu)$  be a finite correlated space. Let  $g : \Omega_2 \rightarrow \mathbb{R}$  be such that  $\mathbb{E}_{(x,y) \sim \mu}[g(y)] = 0$  and  $\mathbb{E}_{(x,y) \sim \mu}[g(y)^2] \leq 1$ . Then, among all functions  $f : \Omega_1 \rightarrow \mathbb{R}$  that satisfy  $\mathbb{E}_{(x,y) \sim \mu}[f(x)^2] \leq 1$ , the maximum value of  $|\mathbb{E}[f(x)g(y)]|$  is given as:*

$$|\mathbb{E}[f(x)g(y)]| = \sqrt{\mathbb{E}_{(x,y) \sim \mu} [(Ug(x))^2]}.$$

► **Proposition B.4** ([14, Proposition 2.11]). *Let  $(\prod_{i=1}^n \Omega_i^{(1)} \times \prod_{i=1}^n \Omega_i^{(2)}, \prod_{i=1}^n \mu_i)$  be a product correlated spaces. Let  $g : \prod_{i=1}^n \Omega_i^{(2)} \rightarrow \mathbb{R}$  be a function and  $U$  be the Markov operator mapping functions from space  $\prod_{i=1}^n \Omega_i^{(2)}$  to the functions on space  $\prod_{i=1}^n \Omega_i^{(1)}$ . If  $g = \sum_{S \subseteq [n]} g_S$  and  $Ug = \sum_{S \subseteq [n]} (Ug)_S$  be the Efron-Stein decomposition of  $g$  and  $Ug$  respectively then,*

$$(Ug)_S = U(g_S)$$

*i.e. the Efron-Stein decomposition commutes with Markov operators.*

► **Proposition B.5** ([14, Proposition 2.12]). *Assume the setting of Proposition B.4 and furthermore assume that  $\rho(\Omega_i^{(1)}, \Omega_i^{(2)}; \mu_i) \leq \rho$  for all  $i \in [n]$ , then for all  $g$  it holds that*

$$\|U(g_S)\|_2 \leq \rho^{|S|} \|g_S\|_2.$$

We will prove the following claim.

► **Claim B.6.** *For each  $i \in [L]$ ,*

$$\rho(\mathcal{X}^i \times \mathcal{Y}_{-k}^i, \mathcal{Y}_k^i; \mathcal{T}_2^i) \leq \sqrt{1 - \varepsilon}.$$

Before proving this claim, first let's see how it leads to the proof of Claim 4.4.

**Proof of Claim 4.4.** Proposition B.4 shows that the Markov operator  $U$  commutes with taking the Efron-Stein decomposition. Hence,  $G_\alpha := (U((I - T_{1-\gamma})f_w))_\alpha = U((I - T_{1-\gamma})(f_w)_\alpha)$ , where  $(f_w)_\alpha$  is the Efron-Stein decomposition of  $f_w$  w.r.t the marginal distribution of  $\mathcal{T}_2$  on  $\prod_{i=1}^L \mathcal{Y}_k^i$  which is a uniform distribution. Therefore,  $(f_w)_\alpha = \sum_{\substack{\beta \subseteq [2R], \\ \tilde{\pi}_{uw}(\beta) = \alpha}} \hat{f}_w(\beta) \chi_\beta$ . Using

Proposition B.5 and Claim B.6, we have

$$\begin{aligned} \|G_\alpha\|_2^2 &= \|U((I - T_{1-\gamma})(f_w)_\alpha)\|_2^2 \leq (\sqrt{1 - \varepsilon})^{2|\alpha|} \|(I - T_{1-\gamma})(f_w)_\alpha\|_2^2 \\ &= (1 - \varepsilon)^{|\alpha|} \sum_{\beta \subseteq [2R]: \tilde{\pi}_{uw}(\beta) = \alpha} \left(1 - (1 - \gamma)^{2|\beta|}\right) \hat{f}_w(\beta)^2, \end{aligned}$$

where the norms are with respect to the marginals of  $\mathcal{T}_2$  in the corresponding spaces. ◀

**Proof of Claim B.6.** Recall the random variable  $c_2 \in \{*, 0, 1\}$  defined in Step 3 of test  $\mathcal{T}_2$ . Let  $g$  and  $f$  be the functions that satisfies  $\mathbb{E}[g] = \mathbb{E}[f] = 0$  and  $\mathbb{E}[g^2], \mathbb{E}[f^2] \leq 1$  such that  $\rho(\mathcal{X}^i \times \mathcal{Y}_{-k}^i, \mathcal{Y}_k^i; \mathcal{T}_2^i) = \mathbb{E}[fg]$ . Define the *Markov Operator*

$$Ug(X^i, Y_{-k}^i) = \mathbb{E}_{(\tilde{X}, \tilde{Y}) \sim \mathcal{T}_2^i} [g(\tilde{Y}_k) \mid (\tilde{X}, \tilde{Y}_{-k}) = (X^i, Y_{-k}^i)].$$

By Lemma B.3, we have

$$\begin{aligned} \rho(\mathcal{X}^i \times \mathcal{Y}_{-k}^i, \mathcal{Y}_k^i; \mathcal{T}_2^i)^2 &\leq \mathbb{E}_{\mathcal{T}_2^i}[Ug(X^i, Y_{-k}^i)^2] \\ &= (1 - 2\varepsilon) \mathbb{E}_{\mathcal{T}_2^i}[Ug(X^i, Y_{-k}^i)^2 \mid c_2 = *] + \varepsilon \mathbb{E}_{\mathcal{T}_2^i}[Ug(X^i, Y_{-k}^i)^2 \mid c_2 = 0] + \\ &\quad \varepsilon \mathbb{E}_{\mathcal{T}_2^i}[Ug(X^i, Y_{-k}^i)^2 \mid c_2 = 1] \\ &\leq (1 - 2\varepsilon) + \varepsilon \mathbb{E}_{\mathcal{T}_2^i}[Ug(X^i, Y_{-k}^i)^2 \mid c_2 = 0] + \varepsilon \mathbb{E}_{\mathcal{T}_2^i}[Ug(X^i, Y_{-k}^i)^2 \mid c_2 = 1], \end{aligned}$$

where the last inequality uses the fact that  $\mathbb{E}_{\mathcal{T}_2^i}[Ug(X^i, Y_{-k}^i)^2 \mid c_2 = *] = \mathbb{E}[g^2]$  which is at most 1. Consider the case when  $c_2 = 0$ . By definition, we have

$$\mathbb{E}_{\mathcal{T}_2^i}[Ug(X^i, Y_{-k}^i)^2 \mid c_2 = 0] = \mathbb{E}_{\substack{X^i, \\ Y_{-k}^i} \sim \mathcal{T}_2^i} \left( \mathbb{E}_{(\tilde{X}, \tilde{Y}) \sim \mathcal{T}_2^i} [g(\tilde{Y}_k) \mid (\tilde{X}, \tilde{Y}_{-k}) = (X^i, Y_{-k}^i) \wedge c_2 = 0] \right)^2.$$

Under the conditioning, for any fixed value of  $X^i, Y_{-k}^i$ , the value of  $\tilde{Y}_k|_{B'(i)}$  is a uniformly random string whereas  $\tilde{Y}_k|_{B(i)}$  is a fixed string (since the *parity* of all columns in  $B(i)$  is 1). Let  $\mathcal{U}$  be the uniform distribution on  $\{-1, +1\}^d$  and  $\mathcal{P}(X^i, Y_{-k}^i) \in \{+1, -1\}^d$  denotes the column wise parities of

$$\begin{bmatrix} X^i|_{B(i)} \\ Y_{-k}^i|_{B(i)} \end{bmatrix}.$$

$$\begin{aligned} \mathbb{E}_{\mathcal{T}_2^i}[Ug(X^i, Y_{-k}^i)^2 \mid c_2 = 0] &= \mathbb{E}_{X^i, Y_{-k}^i \sim \mathcal{T}_2^i} \left( \mathbb{E}_{(\tilde{X}, \tilde{Y}) \sim \mathcal{T}_2^i} [g(\tilde{Y}_k) \mid (\tilde{X}, \tilde{Y}_{-k}) = (X^i, Y_{-k}^i) \wedge c_2 = 0] \right)^2 \\ &= \mathbb{E}_{\substack{X^i, Y_{-k}^i \sim \mathcal{T}_2^i, \\ z = \mathcal{P}(X^i, Y_{-k}^i)}} \left( \mathbb{E}_{r \sim \mathcal{U}} [g(-z, r)] \right)^2 \\ &= \mathbb{E}_{z \sim \mathcal{U}} \left( \mathbb{E}_{r \sim \mathcal{U}} [g(z, r)] \right)^2 \quad (\text{Since marginal on } z \text{ is uniform}) \\ &= \mathbb{E}_{z \sim \mathcal{U}} \left( \mathbb{E}_{r \in \mathcal{U}} \sum_{\alpha \subseteq B(i) \cup B'(i)} \hat{g}(\alpha) \chi_\alpha(z, r) \right)^2 \\ &= \mathbb{E}_{z \sim \mathcal{U}} \left( \sum_{\alpha \subseteq B(i) \cup B'(i)} \hat{g}(\alpha) \mathbb{E}_{r \in \mathcal{U}} [\chi_\alpha(z, r)] \right)^2 \\ &= \mathbb{E}_{z \sim \mathcal{U}} \left( \sum_{\alpha \subseteq B(i)} \hat{g}(\alpha) \chi_\alpha(z) \right)^2 \\ &= \sum_{\alpha \subseteq B(i)} \hat{g}(\alpha)^2. \end{aligned}$$

Similarly we have,

$$\mathbb{E}_{\mathcal{T}_2^i}[Ug(X^i, Y_{-k}^i)^2 \mid c_2 = 1] = \sum_{\alpha \subseteq B'(i)} \hat{g}(\alpha)^2.$$



Now we can bound the correlation as follows:

$$\begin{aligned}
 \rho(\mathcal{X}^i \times \mathcal{Y}_{-k}^i, \mathcal{Y}_k^i; \mathcal{T}_2^i)^2 &\leq (1 - 2\varepsilon) + \varepsilon \sum_{\alpha \subseteq B(i)} \hat{g}(\alpha)^2 + \varepsilon \sum_{\alpha \subseteq B'(i)} \hat{g}(\alpha)^2 \\
 &\leq (1 - 2\varepsilon) + \varepsilon \sum_{\alpha \subseteq B(i) \cup B'(i)} \hat{g}(\alpha)^2 \quad (\text{Using } \hat{g}(\phi) = \mathbb{E}[g] = 0) \\
 &\leq (1 - \varepsilon). \quad (\text{Using } \mathbb{E}[g^2] \leq 1 \text{ and Parseval's Identity})
 \end{aligned}$$



# Subexponential Size Hitting Sets for Bounded Depth Multilinear Formulas\*

Rafael Oliveira<sup>1</sup>, Amir Shpilka<sup>2</sup>, and Ben Lee Volk<sup>2</sup>

- 1 Department of Computer Science,  
Princeton University, USA  
rmo@cs.princeton.edu
- 2 Department of Computer Science  
Tel Aviv University, Israel  
shpilka@post.tau.ac.il, benleevolk@gmail.com

---

## Abstract

In this paper we give subexponential size hitting sets for bounded depth multilinear arithmetic formulas. Using the known relation between black-box PIT and lower bounds we obtain lower bounds for these models.

For depth-3 multilinear formulas, of size  $\exp(n^\delta)$ , we give a hitting set of size  $\exp(\tilde{O}(n^{2/3+2\delta/3}))$ . This implies a lower bound of  $\exp(\tilde{\Omega}(n^{1/2}))$  for depth-3 multilinear formulas, for some explicit polynomial.

For depth-4 multilinear formulas, of size  $\exp(n^\delta)$ , we give a hitting set of size  $\exp(\tilde{O}(n^{2/3+4\delta/3}))$ . This implies a lower bound of  $\exp(\tilde{\Omega}(n^{1/4}))$  for depth-4 multilinear formulas, for some explicit polynomial.

A regular formula consists of alternating layers of  $+$ ,  $\times$  gates, where all gates at layer  $i$  have the same fan-in. We give a hitting set of size (roughly)  $\exp(n^{1-\delta})$ , for regular depth- $d$  multilinear formulas of size  $\exp(n^\delta)$ , where  $\delta = O(\frac{1}{\sqrt{5}^d})$ . This result implies a lower bound of roughly  $\exp(\tilde{\Omega}(n^{\frac{1}{\sqrt{5}^d}}))$  for such formulas.

We note that better lower bounds are known for these models, but also that none of these bounds was achieved via construction of a hitting set. Moreover, no lower bound that implies such PIT results, even in the white-box model, is currently known.

Our results are combinatorial in nature and rely on reducing the underlying formula, first to a depth-4 formula, and then to a read-once algebraic branching program (from depth-3 formulas we go straight to read-once algebraic branching programs).

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

**Keywords and phrases** Arithmetic Circuits, Derandomization, Polynomial Identity Testing

**Digital Object Identifier** 10.4230/LIPIcs.CCC.2015.304

## 1 Introduction

Arithmetic circuits are the standard model for computing polynomials. Roughly speaking, given a set of variables  $X = \{x_1, \dots, x_n\}$ , an arithmetic circuit uses additions and multiplications to compute a polynomial  $f$  in the set of variables  $X$ . An arithmetic formula is an

---

\* The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement number 257575, from the Israel Science Foundation (grant number 339/10), from NSF grant CCF-1217416 and from the Sloan fellowship.

arithmetic circuit whose computation graph is a tree. An arithmetic circuit (or formula) is multilinear if the polynomial computed at each of its gates is multilinear (as a formal polynomial), that is, in each of its monomials the power of every input variable is at most one (see Section 1.1 for definition of the models studied in this paper)

Two outstanding open problems in complexity theory are to prove exponential lower bounds on the size of arithmetic circuits, i.e., to prove a lower bound on the number of operations required to compute some polynomial  $f$ , and to give efficient deterministic polynomial identity testing (PIT for short) algorithms for them. The PIT problem for arithmetic circuits asks the following question: given an arithmetic circuit  $\Phi$  computing a polynomial  $f$ , determine, *efficiently and deterministically*, whether “ $f \equiv 0$ ”. The black-box version of the PIT problem asks to construct a small *hitting set*, i.e., a set of evaluation points  $\mathcal{H}$ , for which any such non-zero  $f$  does not vanish on all the points in  $\mathcal{H}$ .

It is known that solving any one of the problems (proving lower bound or deterministic PIT), with appropriate parameters, for small depth (multilinear) formulas, is equivalent to solving it in the general (multilinear) case [37, 6, 24, 15, 36]. It is also known that these two problems are tightly connected and that solving one would imply a solution to the other, both in the general case [16, 17, 1] and in the bounded depth case<sup>1</sup> [11]. We note that in the multilinear case it is only known that hitting sets imply circuit lower bounds but not vice versa.

In this work we study the PIT problem for several models of bounded depth multilinear formulas. Our main results are subexponential size hitting sets for depth-3 and depth-4 multilinear formulas of subexponential size and for *regular* depth- $d$  multilinear formulas of subexponential size (with construction size deteriorating among the different models). Using the connection between explicit hitting sets and circuit lower bounds we get, as corollaries, subexponential lower bounds for these models.

## 1.1 Models for Computing Multilinear Polynomials

An arithmetic circuit  $\Phi$  over the field  $\mathbb{F}$  and over the set of variables  $X$  is a directed acyclic graph as follows. Every vertex in  $\Phi$  of in-degree 0 is labelled by either a variable in  $X$  or a field element in  $\mathbb{F}$ . Every other vertex in  $\Phi$  is labelled by either  $\times$  or  $+$ . An arithmetic circuit is called a formula if it is a directed tree (whose edges are directed from the leaves to the root). The vertices of  $\Phi$  are also called gates. Every gate of in-degree 0 is called an input gate. Every gate of out-degree 0 is called an output gate. Every gate labelled by  $\times$  is called a product gate. Every gate labelled by  $+$  is called a sum gate. An arithmetic circuit computes a polynomial in a natural way. An input gate labelled by  $y \in \mathbb{F} \cup X$  computes the polynomial  $y$ . A product gate computes the product of the polynomials computed by its children. A sum gate computes the sum of the polynomials computed by its children.

A polynomial  $f \in \mathbb{F}[X]$  is called multilinear if the degree of each variable in  $f$  is at most one. An arithmetic circuit (formula)  $\Phi$  is called multilinear if every gate in  $\Phi$  computes a multilinear polynomial.

In this work we are interested in small depth multilinear formulas. A depth-3  $\Sigma\Pi\Sigma$  formula is a formula composed of three layers of alternating sum and product gates. Thus,

---

<sup>1</sup> The result of [11] is more restricted than the results for circuits with no depth restrictions.

every polynomial computed by a  $\Sigma\Pi\Sigma$  formula of size  $s$  has the following form

$$f = \sum_{i=1}^s \prod_{j=1}^{d_i} \ell_{i,j},$$

where the  $\ell_{i,j}$  are linear functions. In a  $\Sigma\Pi\Sigma$  multilinear formula, it holds that in every product gate,  $\prod_{j=1}^{d_i} \ell_{i,j}$ , the linear functions  $\ell_{i,1}, \dots, \ell_{i,d_i}$  are supported on disjoint sets of variables.

Similarly, a depth-4  $\Sigma\Pi\Sigma\Pi$  formula is a formula composed of four layers of alternating sum and product gates. Thus, every polynomial computed by a  $\Sigma\Pi\Sigma\Pi$  formula of size  $s$  has the following form

$$f = \sum_{i=1}^s \prod_{j=1}^{d_i} Q_{i,j},$$

where the  $Q_{i,j}$  are computed at the bottom  $\Sigma\Pi$  layers and are  $s$ -sparse polynomials, i.e., polynomials that have at most  $s$  monomials. As in the depth-3 case, we have that at every product gate the polynomials  $Q_{i,1}, \dots, Q_{i,d_i}$  are supported on disjoint sets of variables.

Another important definition for us is that of a regular depth- $d$  formula. A regular depth- $d$  formula is specified by a list of  $d$  integers  $(a_1, p_1, a_2, p_2, \dots)$ . It has  $d$  layers of alternating sum and product gates. The fan-in of every sum gate at the  $(2i - 1)$ 'th layer is  $a_i$  and, similarly, the fan-in of every product gate at the  $(2i)$ 'th layer is  $p_i$ . For example, a depth-4 formula that is specified by the list  $(a_1, p_1, a_2, p_2)$  has the following form:

$$f = \sum_{i=1}^{a_1} \prod_{j=1}^{p_1} Q_{i,j},$$

where each  $Q_{i,j}$  is a polynomial of degree  $p_2$  that has (at most)  $a_2$  monomials. As before, a regular depth- $d$  multilinear formula is a regular depth- $d$  formula in which every gate computes a multilinear polynomial.

Regular formulas were first defined by Kayal, Saha and Saptharishi [21], who proved quasi-polynomial lower bounds for logarithmic-depth regular formulas. It is interesting to note that in the reductions from general (multilinear) circuits/formulas to depth- $d$  (multilinear) formulas, one gets a regular depth- $d$  (multilinear) formula [37, 6, 24, 36].

Finally, we also need to consider the model of Read-Once Algebraic Branching Programs (ROABPs) as our construction is based on a reduction to this case. Algebraic branching programs were first defined in the work of Nisan [25] who proved exponential lower bounds on the size of non-commutative ABPs computing the determinant or permanent polynomials. Roughly, an algebraic branching program (ABP) consists of a layered graph with edges going from the  $i$ 'th layer to the  $(i + 1)$ 'th layer. The first layer consists of a single source node and the last layer contains a single sink. The edges of the graph are labeled with polynomials (in our case we only consider linear functions as labels). The weight of a path is the product of the weights of the edges in the path. The polynomial computed by the ABP is the sum of the weights of all the paths from the source to the sink. An ABP is called a read-once ABP (ROABP) if the only variable appearing on edges that connect the  $i$ 'th and the  $(i + 1)$ 'th layer is  $x_i$ . It is clear that a ROABP whose edges are labeled with linear functions computes a multilinear polynomial.

## 1.2 Polynomial Identity Testing

In the PIT problem we are given an arithmetic circuit or formula  $\Phi$ , computing some polynomial  $f$ , and we have to determine whether " $f \equiv 0$ ". That is, we are asking if  $f$  is the

zero polynomial in  $\mathbb{F}[x_1, \dots, x_n]$ . By the Schwartz-Zippel-DeMillo-Lipton lemma [38, 32, 9], if  $0 \neq f \in \mathbb{F}[x_1, \dots, x_n]$  is a polynomial of degree  $\leq d$ , and  $\alpha_1, \dots, \alpha_n \in A \subseteq \mathbb{F}$  are chosen uniformly at random, then  $f(\alpha_1, \dots, \alpha_n) = 0$  with probability at most<sup>2</sup>  $d/|A|$ . Thus, given  $\Phi$ , we can perform these evaluations efficiently, giving an efficient randomized procedure for answering “ $f \equiv 0$ ?”. It is an important open problem to find a derandomization of this algorithm, that is, to find a *deterministic* procedure for PIT that runs in polynomial time (in the size of  $\Phi$ ).

One interesting property of the above randomized algorithm of Schwartz-Zippel is that the algorithm does not need to “see” the circuit  $\Phi$ . Namely, the algorithm only uses the circuit to compute the evaluations  $f(\alpha_1, \dots, \alpha_n)$ . Such an algorithm is called a *black-box* algorithm. In contrast, an algorithm that can access the internal structure of the circuit  $\Phi$  is called a *white-box* algorithm. Clearly, the designer of the algorithm has more resources in the white-box model and so one can expect that solving PIT in this model should be a simpler task than in the black-box model.

The problem of derandomizing PIT has received a lot of attention in the past few years. In particular, many works examine a specific class of circuits  $\mathcal{C}$ , and design PIT algorithms only for circuits in that class. One reason for this attention is the strong connection between deterministic PIT algorithms for a class  $\mathcal{C}$  and lower bounds for  $\mathcal{C}$ . This connection was first observed by Heintz and Schnorr [16] (and later also by Agrawal [1]) for the black-box model and by Kabanets and Impagliazzo [18] for the white-box model (see also Dvir, Shpilka and Yehudayoff [11] for a similar result for bounded depth circuits). Another motivation for studying the problem is its relation to algorithmic questions. Indeed, the famous deterministic primality testing algorithm of Agrawal, Kayal and Saxena [3] is based on derandomizing a specific polynomial identity. Finally, the PIT problem is, in some sense, the most general problem that we know today for which we have randomized coRP algorithms but no polynomial time algorithms, thus studying it is a natural step towards a better understanding of the relation between RP and P. For more on the PIT problem we refer to the survey by Shpilka and Yehudayoff [35].

Among the most studied circuit classes we find Read-Once Algebraic Branching Programs [14, 12, 2], set-multilinear formulas [28, 13, 5], depth-3 formulas [10, 23, 20, 22, 31], multilinear depth-4 formulas (and some generalizations of them) [19, 30, 4] and bounded-read multilinear formulas [33, 34, 8, 4]. We note that none of these results follow from a reduction to a la Kabanets-Impagliazzo [18] (or the reduction of [11] for bounded depth circuits) from PIT to lower bounds. Indeed, this reduction does not work for the restricted classes mentioned here. In particular, for the multilinear model no reduction from PIT to lower bounds is known. That is, even given lower bounds for multilinear circuits/formulas (e.g., the exponential lower bound of Raz and Yehudayoff [29] for constant depth multilinear formulas) we do not know how to construct a PIT algorithm for a related model.

The works on depth-3 and multilinear depth-4 formulas gave polynomial time algorithms only when the fan-in of the top gate (the output gate) is constant, and became exponential time when the top fan-in was  $\Omega(n)$ , both in the white-box and black-box models [23, 31, 30]. Raz and Shpilka [28] gave a polynomial time PIT for set-multilinear depth-3 circuits and Forbes and Shpilka [14] and Agrawal, Saha and Saxena [5] gave a quasi-polynomial size hitting set for the model. Recall that in a depth-3 set-multilinear formula, the variables are partitioned to sets, and each linear function at the bottom layer only involves variables

<sup>2</sup> Note that this is meaningful only if  $d < |A| \leq |\mathbb{F}|$ , which in particular implies that  $f$  is not the zero function.

from a single set. Recently, Agrawal et al. [2] gave a subexponential white-box algorithm for a depth-3 formula that computes the sum of  $c$  set-multilinear formulas, each of size  $s$ , with respect to different partitions of the variables. The running time of their algorithm is  $n^{O(2^c n^{1-\frac{2}{2^c}} \log s)}$ . In particular, for  $c = O(\log \log(n))$  the running time is  $\exp(n)$ .

Thus, prior to this work there were no subexponential PIT algorithms, even for depth-3 multilinear formulas with top fan-in  $n$ .

### 1.3 Our Results

► **Remark.** *Throughout this paper, we assume that for formulas of size  $2^{n^\delta}$ , the underlying field  $\mathbb{F}$  is of size at least  $|\mathbb{F}| \geq 2^{n^{2^\delta \text{poly} \log(n)}}$ , and that if this is not the case then we are allowed to query the formula on inputs from an extension field of the appropriate size. In particular, all our results hold over fields of characteristic zero or over fields of size  $\exp(n)$ .*

We give subexponential size hitting sets for depth-3, depth-4 and regular depth- $d$  multilinear formulas, of subexponential size. In particular we obtain the following results.

► **Theorem 1.1.** *There exists a hitting set  $\mathcal{H}$  of size  $|\mathcal{H}| = 2^{\tilde{O}(n^{\frac{2}{3} + \frac{2}{3}\delta})}$  for the class of  $\Sigma\Pi\Sigma$  multilinear formulas of size  $2^{n^\delta}$ .*

This gives a significant improvement to the recent result, mentioned above, of Agrawal et al. [2] who studied sum of set-multilinear formulas. From the connection between hitting sets and circuit lower bounds [16, 1] we obtain the following corollary.

► **Corollary 1.2.** *There is an explicit multilinear polynomial  $f \in \mathbb{F}[x_1, \dots, x_n]$ , whose coefficients can be computed in exponential time, such that any depth-3 multilinear formula for  $f$  has size  $\exp(\tilde{\Omega}(\sqrt{n}))$ .*

This lower bound is weaker than the exponential lower bound of Nisan and Wigderson for this model [26]. Yet, it is interesting to note that we can get such a strong lower bound from a PIT algorithm. Next, we present our result for depth-4 multilinear formulas.

► **Theorem 1.3.** *There exists a hitting set  $\mathcal{H}$  of size  $|\mathcal{H}| = 2^{\tilde{O}(n^{2/3+4\delta/3})}$  for the class of  $\Sigma\Pi\Sigma\Pi$  multilinear formulas of size  $2^{n^\delta}$ .*

Again, from the connection between hitting sets and circuit lower bounds we obtain the following corollary.

► **Corollary 1.4.** *There is an explicit multilinear polynomial  $f \in \mathbb{F}[x_1, \dots, x_n]$ , whose coefficients can be computed in exponential time, such that any depth-4 multilinear formula for  $f$  has size  $\exp(\tilde{\Omega}(n^{1/4}))$ .*

The best known lower bound for depth-4 multilinear formulas is  $\exp(n^{1/2})$  due to Raz and Yehudayoff [29], thus, as in the previous case, the term in the exponent of our lower bound is the square root of the corresponding term in the best known lower bound. For regular depth- $d$  multilinear formulas we obtain the following result.

► **Theorem 1.5.** *There exists a hitting set  $\mathcal{H}$  of size  $|\mathcal{H}| = 2^{\tilde{O}(n^{1-\delta/3})}$  for the class of regular depth- $d$  multilinear formulas of size  $2^{n^\delta}$ , where  $\delta \leq \frac{1}{5^{\lfloor d/2 \rfloor + 1}} = O\left(\frac{1}{\sqrt{5^d}}\right)$ .*

As before we obtain a lower bound for such formulas.

► **Corollary 1.6.** *There is an explicit multilinear polynomial  $f \in \mathbb{F}[x_1, \dots, x_n]$ , whose coefficients can be computed in exponential time, such that any regular depth- $d$  multilinear formula for  $f$  has size  $\exp(\tilde{\Omega}(n^{\frac{1}{5^{\lfloor d/2 \rfloor + 1}}}))$ .*

We note that Raz and Yehudayoff gave an  $\exp(n^{\Omega(\frac{1}{d})})$  lower bound for depth- $d$  multilinear formulas, which is much stronger than what Corollary 1.6 gives. Yet, our result also gives a PIT algorithm, which does not follow from the results of [29]. As we later explain, we lose a square root in the term at the exponent for every increase of the depth and this is the reason that we get only  $\exp(n^{1/\exp(d)})$  instead of  $\exp(n^{1/d})$ .

## 1.4 Proof Overview

We first discuss our proof technique for the case of depth-3 multilinear formulas. Our (idealized) aim is to reduce such a formula  $\Phi$  to a depth-3 multilinear formula in which each linear function is of the form  $\alpha x + \beta$ . That is, each linear function contains at most one variable. If we manage to do that then we can use the quasi-polynomial sized hitting set of [13, 2] for this model.

Of course, the problem with the above argument is that in general, depth-3 formulas have more than one variable per linear function. To overcome this difficulty, we will partition the variables to several sets  $T_1, \dots, T_m$  and hope that each linear function in the formula contains at most one variable from each  $T_i$ . If we can do that then we would use the hitting set for each set of variables  $T_i$  and combine those sets together to get our hitting set. That is, the combined hitting set is composed of concatenation of all vectors of the form  $v_1 \circ v_2 \circ \dots \circ v_m$  where  $v_i$  comes from the hitting set restricted to the variables in  $T_i$  (the concatenation is performed in a way that respects the partition of course). Thus, if we can carry out this procedure then we will get a hitting set of size roughly  $n^{m \log n}$ . This step indeed yields a hitting set, since when we restrict our attention to each  $T_i$  and think of the other variables as constants in some huge extension field, then we do get a small ROABP (in the variables of  $T_i$ ) and hence plugging in the hitting set of [13, 2] gives a non-zero polynomial. Thus, we can first do this for  $T_1$  and obtain some good assignment  $v_1$  that makes the polynomial non-zero after substituting  $v_1$  to  $T_1$ . Then we can find  $v_2$  etc.

There are two problems with the above argument. One problem is how to find such a good partition. The second is that this idea simply cannot work as is. For example, if we have the linear function  $x_1 + \dots + x_n$ , then it will have a large intersection with each  $T_i$ .

We first deal with the second question. to overcome the difficulty posed by the example, we would like to somehow “get rid” of all linear functions of large support and then carry out the idea above. To remove linear functions with large support from the formula we use another trick. Consider a variable  $x_k$  that appears in a linear function  $\ell_0$  that has a large support. Assume that  $\frac{\partial f}{\partial x_k} \neq 0$  as otherwise we can ignore  $x_k$ . Now, because of multilinearity, we can transform our original formula  $\Phi$  to a formula computing  $\frac{\partial f}{\partial x_k}$ . This is done by replacing each linear function  $\ell(X) = \sum_{i=1}^n \alpha_i x_i + \alpha_0$  with the constant  $\alpha_k$ . In particular, the function  $\ell_0$  that used to have a high support does not appear in the new formula. Furthermore, this process does not increase the support size of any other linear function. A possible issue is that if we have to repeat this process for every function of large support then it seems that we need to take a fresh derivative for every such linear function. The point is that because we only care about linear functions that have a large support to begin with, we can find a variable that simultaneously appears in many of those functions and thus one derivative will eliminate many of the “bad” linear functions. Working out the parameters, we see that we need to take roughly  $n^\epsilon \cdot \log |\Phi|$  many derivatives to reduce to the case where all linear functions have support size at most  $n^{1-\epsilon}$ .

Now we go back to our first problem. We can assume that we have a depth-3 formula in which each linear function has support size at most  $n^{1-\epsilon}$  and we wish to find a partition of the variables to sets  $T_1, \dots, T_m$  so that each  $T_i$  contains at most one variable from each

linear function. This cannot be achieved as a simple probabilistic argument shows, so we relax our requirement and only demand that in each multiplication gate (of the formula) only a few linear functions have a large intersection. If at most  $k$  linear functions in each gate have a large intersection, we can expand each multiplication gate to at most  $n^k$  new gates (by simply expanding all linear functions that have large intersection) and then apply our argument. As we will be able to handle subexponential size formulas, this blow up is tolerable for us.

Note that if we were to pick the partition at random, when  $m = n^{1-\epsilon+\gamma}$ , for some small  $\gamma$ , then we will get that with very high probability at most  $n^\delta$  linear functions will have intersection at most  $n^\delta$  with each  $T_i$ , where  $\delta$  is such that  $|\Phi| < \exp(n^\delta)$ . To get a deterministic version of this partitioning, we simply use an  $n^\delta$ -wise independent family of hash functions  $\{h : [n] \rightarrow [m]\}$ . Each hash function  $h$  induces a partition of the variables to  $T_i = \{x_k \mid h(k) = i\}$ . Because of the high independence, we are guaranteed that there is at least one hash function that induces a good partition.

Now we have all the ingredients in place. To get our hitting set we basically do the following (we describe the construction as a process, but it should be clear that every step can be performed using some evaluation vectors).

1. Pick  $n^\epsilon \cdot \log |\Phi|$  many variables and compute a black-box for the polynomial that is obtained by taking the derivative of  $f$  with respect to those variables. The cost of this step is roughly  $\binom{n}{n^\epsilon \cdot \log |\Phi|} \cdot 2^{n^\epsilon \cdot \log |\Phi|}$ , where the first term is for picking the variables and the second is what we have to pay to get access to the derived polynomial.
2. Partition the remaining variables to (roughly)  $n^{1-\epsilon/2}$  many sets using a (roughly)  $\log |\Phi|$ -wise independent family of hash functions. The cost of this step is roughly  $n^{\log |\Phi|}$  as this is the size of the hash function family.
3. Plug in a fresh copy of the hitting set of [13, 2] to each of the sets of variables  $T_i$ . The cost is roughly  $n^{\log n \cdot n^{1-\epsilon/2}}$ .

Combining everything we get a hitting set of size roughly

$$\left( \binom{n}{n^\epsilon \cdot \log |\Phi|} \cdot 2^{n^\epsilon \cdot \log |\Phi|} \right) \cdot \left( n^{\log |\Phi|} \right) \cdot \left( n^{\log n \cdot n^{1-\epsilon/2}} \right) \approx 2^{\tilde{O}(n^{1-\epsilon/2} + n^\epsilon \log |\Phi|)}.$$

Optimizing the parameters we get our hitting set.

We would like to use the same approach also for the case of depth-4 formulas. Here the problem is that in the two bottom layers the formula computes a polynomial and not a linear function. In particular, when taking a derivative we are no longer removing functions that have large support. Nevertheless, we can still use a similar idea. We show there is a variable  $x_i$  that by either setting  $f|_{x_i=0}$  or considering  $\frac{\partial f}{\partial x_i}$ , we are guaranteed that the total sparsity of all polynomials that have large support goes down by some non-negligible factor. Thus, repeating this process (of either setting a variable to 0 or taking a derivative)  $n^\epsilon \cdot \log |\Phi|$  many times we reach a depth-4 formula where all polynomials computed at the bottom addition gate have small support. Next, we partition the variables to sets and consider a single set  $T_i$ . Now, another issue is that even if the intersection of a low-support polynomial with some  $T_i$  is rather small, the sparsity of the resulting polynomial (which is considered as a polynomial in the variable in the intersection) can still be exponential in the size of the intersection. This is why we lose a bit in the upper bound compared to the depth-3 case. Combining all steps again we get the result for depth-4 formulas.

The proof for regular formulas works by first reducing to the depth-4 case and then applying our hitting set. The reduction is obtained in a similar spirit to the reduction of Kayal et al. [21]. We break the formula at an appropriate layer and then express the top



layers as a  $\Sigma\Pi$  circuit and the bottom layers as products of polynomials of not too high degree. We then use the trivial observation that if the degree of a polynomial is at most  $n^{1-\epsilon}$  then its sparsity is at most  $n^{1-\epsilon}$  and proceed as before. Due to the different requirements of the reduction and of the hashing part, we roughly lose a constant factor in the exponent of  $n$ , in the size of the hitting set, whenever the depth grows, resulting in a hitting set of size roughly  $\exp(n^{1-1/\exp(d)})$ .

To obtain the lower bounds we simply use the idea of [16, 1]. That is, given a hitting set we find a non-zero multilinear polynomial that vanishes on all points of the hitting set by solving a homogeneous system of linear equations.

## 1.5 Related Work

### 1.5.1 The work of Agrawal, Gurjar, Korwar and Saxena [2]

The closest work to ours is the one by Agrawal et al. [2]. In addition to other results, they gave a white-box PIT algorithm that runs in time  $n^{O(2^c n^{1-\frac{2}{2^c}} \log s)}$  for depth-3 formulas that can be represented as a sum of  $c$  set-multilinear formulas, each of size  $s$  (potentially with respect to different partitions of the variables).

Theorem 1.1 improves upon this results in several ways. First, the theorem gives a hitting set, i.e., a black-box PIT. Secondly, for  $c = O(\log \log n)$  the running time of the algorithm of [2] is  $\exp(n)$ , whereas our construction can handle a sum of  $\exp(n^\beta)$  set-multilinear formulas and still maintain a subexponential complexity.

Nonetheless, there are some similarities behind the basic approach of this work and the work of Agrawal et al. Recall that a set-multilinear depth-3 formula is based on a partition of the variables, where each linear function in the formula contains variables from a single partition. Agrawal et al. start with a sum of  $c$  set-multilinear circuits, each with respect to a different partitioning of the variables, and their first goal is to reduce the formula to a set-multilinear formula, i.e., to have only one partition of the variables. For this they define a distance between different partitions and show, using an involved combinatorial argument, that one can find some partition  $T_1, \dots, T_m$  of the variables so that when restricting our attention to  $T_i$ , all the  $c$  set-multilinear formulas will be somewhat “close to each other”. If the distance is  $\Delta$  (according to their definition) then they prove that they can express the sum as a set-multilinear circuit of size roughly  $s \cdot n^\Delta$ , where  $s$  is the total size of the depth-3 formula. Unlike our work, they find the partition in a white-box manner by gradually refining the given  $c$  partitions of the set-multilinear circuits composing the formula. The final verification step is done, in a similar manner to ours, by substituting the hitting set of [5] (or that of [13]) to each of the sets  $T_i$ . The step of finding the partition  $T_1, \dots, T_m$  is technically involved and is the only step where white-box access is required.

## 1.6 Organization

In Section 2 we provide basic definitions and notations, and also state some general lemmas which will be helpful in the next sections. In Section 3, we explain how to reduce general depth-3 and depth-4 formulas to formulas such that every polynomial at the bottom has small support. Then, in Section 4, we construct a hitting set for those types of formulas. In Section 5, we explain how to combine the ideas of the previous two sections and construct our hitting set for depth-3 and depth-4 multilinear formulas.

We then move on to depth- $d$  regular formulas in Section 6, and state our hitting set for this class. In the short Section 7 we spell out briefly how, using known observations about the

relation between PIT and lower bounds, we obtain our lower bounds for multilinear formulas. Finally, in Section 8 we discuss some open problems and future directions for research.

The proofs of the results regarding depth-4 and depth- $d$  regular formulas are omitted from this version, and can be found in the full version of the paper ([27]).

## 2 Preliminaries

In this section, we establish notation, some definitions and useful lemmas that will be used throughout the paper.

### 2.1 Notations and Basic Definitions

For any positive integer  $n$ , we denote by  $[n]$  the set of integers from 1 to  $n$ , and by  $\binom{[n]}{\leq r}$  the family of subsets  $A \subseteq [n]$  such that  $|A| \leq r$ . We often associate a subset  $A \subseteq [n]$  with a subset of variables  $\text{var}(A) \subseteq \{x_1, \dots, x_n\}$  in a natural way (i.e.,  $\text{var}(A) = \{x_i \mid i \in A\}$ ). In those cases we make no distinction between the two and use  $A$  to refer to  $\text{var}(A)$ . Additionally, if  $A$  and  $B$  are disjoint subsets of  $[n]$ , we denote their disjoint union by  $A \sqcup B$ . For a vector  $v \in \mathbb{F}^n$  we denote with  $v|_A$  the restriction of  $v$  to the coordinates  $A$ . In order to improve the readability of the text, we omit floor and ceiling notations.

Let  $f(x_1, \dots, x_n) \in \mathbb{F}[x_1, \dots, x_n]$  be a polynomial. We will denote by  $\partial_x f$  the formal derivative of  $f$  with respect to the variable  $x$ , and by  $f|_{x=0}$  the polynomial obtained from  $f$  by setting  $x = 0$ . Moreover, if  $A \subseteq [n]$ , we will denote by  $\partial_A f$  the polynomial obtained when taking the formal derivative of  $f$  with respect to all variables in  $A$ . In a similar fashion, we denote by  $f|_{A=0}$  the polynomial obtained when we set all the variables in  $A$  to zero, and more generally, if  $|A| = r$  and  $\bar{\alpha} = (\alpha_1, \dots, \alpha_r) \in \mathbb{F}^r$ ,  $f|_{A=\bar{\alpha}}$  will denote the restriction of  $f$  obtained when setting the  $i$ 'th variable in  $A$  to  $\alpha_i$ , for  $1 \leq i \leq r$ .

In addition to the conventions above, the following definitions will be very useful in the next sections.

► **Definition 2.1** (Variable Set and Non-trivial Variable Set). Let  $f(x_1, \dots, x_n) \in \mathbb{F}[x_1, \dots, x_n]$  be a polynomial. Define the variable set ( $\text{var}$ ) and the non-trivial variable set ( $\text{var}^*$ ) as follows:

$$\begin{aligned} \text{var}(f) &= \{x \in \{x_1, \dots, x_n\} \mid \partial_x f \neq 0\} \\ \text{var}^*(f) &= \{x \in \{x_1, \dots, x_n\} \mid \partial_x f \neq 0 \text{ and } f|_{x=0} \neq 0\}. \end{aligned}$$

That is, the variable set of a polynomial  $f$  is the set of variables  $x \in \{x_1, \dots, x_n\}$  that appear in the representation of  $f$  as a sum of monomials, whereas the non-trivial variable set is the set of variables of  $f$  that do not divide it.

We shall say that  $f$  has a small support if  $\text{var}(f)$  (or  $\text{var}^*(f)$ ) is not too large.

► **Definition 2.2** (Monomial Support and Sparsity). Let  $f(x_1, \dots, x_n) \in \mathbb{F}[x_1, \dots, x_n]$  be a polynomial. We define the *monomial support* of  $f$ , written  $\text{mon}(f)$ , as the set of monomials that have a non-zero coefficient in  $f$ . In addition, we define the sparsity of  $f$ , written  $\|f\|$ , as the size of the set  $\text{mon}(f)$ , that is,

$$\|f\| = |\text{mon}(f)|.$$

In other words, the sparsity of  $f$  is the number of monomials that appear with a non-zero coefficient in  $f$ .

In the constructions of our hitting sets we will need to combine assignments to different subsets of variables. The following notation will be useful. For a partition of  $[n]$ ,  $T_1 \sqcup T_2 \sqcup \dots \sqcup T_m = [n]$ , and sets  $\mathcal{H}_i \subseteq \mathbb{F}^{T_i}$ , we denote with  $\mathcal{H}_1^{T_1} \times \dots \times \mathcal{H}_m^{T_m}$  the set of all vectors of length  $n$  whose restriction to  $T_i$  is an element of  $\mathcal{H}_i$ :

$$\mathcal{H}_1^{T_1} \times \dots \times \mathcal{H}_m^{T_m} = \{v \in \mathbb{F}^n \mid \forall i \in [m], v|_{T_i} \in \mathcal{H}_i\}.$$

## 2.2 Depth-3 and Depth-4 Formulas

We define some special classes of depth-3 and depth-4 formulas that will be used throughout this paper.

► **Definition 2.3** (Restricted Top Fan-in). Let  $\Phi$  be a multilinear depth-4 formula. We say that  $\Phi$  is a multilinear  $\Sigma^{[M]}\Pi\Sigma\Pi$  formula if it is of the form  $\sum_{i=1}^m \prod_{j=1}^{t_i} f_{i,j}$ , where  $m \leq M$ . If, in addition to the conditions above, we have that each  $f_{i,j}$  is a linear function, that is,  $\Phi$  is actually a depth-3 formula, we will say that  $\Phi$  is a multilinear  $\Sigma^{[M]}\Pi\Sigma$  formula.

Our next definition considers the case where polynomials computed at the bottom layers do not contain too many variables, that is, they have small variable set.

► **Definition 2.4** (Restricted Top Fan-in and Variable Set). Let  $\Phi$  be a multilinear depth-4 formula. We say that  $\Phi$  is a multilinear  $\Sigma^{[M]}\Pi(\Sigma\Pi)^{\{\tau\}}$  formula if it is of the form  $\sum_{i=1}^m \prod_{j=1}^{t_i} f_{i,j}$ , where  $m \leq M$  and for each  $1 \leq i \leq m$  we have that

1.  $|\text{var}(f_{i,j})| \leq \tau$  for all  $1 \leq j \leq t_i$
2.  $\text{var}(f_{i,j_1}) \cap \text{var}(f_{i,j_2}) = \emptyset$ , for any  $j_1 \neq j_2$ .

If, in addition to the conditions above, we have that each  $f_{i,j}$  is a linear function, that is,  $\Phi$  is actually a depth-3 formula, we will say that  $\Phi$  is a multilinear  $\Sigma^{[M]}\Pi\Sigma^{\{\tau\}}$  formula.

Since the formula will be given to us as a black-box, we can make some assumptions about it, which will help us to preserve non-zerosness when taking derivatives or setting variables to zero. To this end, we define a notion of simplicity of depth-4 formulas,<sup>3</sup> and prove that we can assume without loss of generality that any input formula is simple.

► **Definition 2.5.** Let  $f(x_1, \dots, x_n) \in \mathbb{F}[x_1, \dots, x_n]$  be a multilinear polynomial and let

$$\Phi = \sum_{i=1}^M \prod_{j=1}^{t_i} f_{i,j}$$

be a multilinear depth-4 formula computing  $f$ . We say that  $\Phi$  is a *simple* multilinear depth-4 formula if for each variable  $x \in \text{var}(f)$  that divides  $f$ , it must be the case that for every  $1 \leq i \leq M$ , there exists  $j \in [t_i]$  such that  $f_{i,j} = x$ .

In words,  $\Phi$  is simple if whenever a variable  $x$  divides  $f$ , it also divides every product gate. The following proposition tells us that we can indeed assume, without loss of generality, that any multilinear depth-4 formula given to us is a simple formula.

► **Proposition 2.6.** *If  $\Phi$  is a depth-4 multilinear  $\Sigma^{[M]}\Pi\Sigma\Pi$  formula computing  $f(x_1, \dots, x_n)$ , then  $f$  can be computed by a simple depth-4 multilinear  $\Sigma^{[M]}\Pi\Sigma\Pi$  formula  $\Psi$  where  $|\Psi| \leq |\Phi|$ .*

<sup>3</sup> Note that this is not the same notion as used, e.g., in [10].

As a corollary, together with the simple observation that any derivative or restriction of a multilinear formula results in a multilinear formula of at most the same size, we obtain that partial derivatives or restrictions of a multilinear polynomial can also be computed by simple formulas.

► **Corollary 2.7.** *If  $\Phi$  is a depth-4 multilinear  $\Sigma^{[M]}\Pi\Sigma\Pi$  formula computing  $f(x_1, \dots, x_n)$ , then for any disjoint sets  $A, B \subseteq \text{var}(f)$ ,  $\partial_A f|_{B=0}$  can be computed by a simple depth-4 multilinear  $\Sigma^{[M]}\Pi\Sigma\Pi$  formula  $\Psi$  where  $|\Psi| \leq |\Phi|$ . We will refer to  $\Psi$  as  $\partial_A \Phi|_{B=0}$ .*

Therefore, from now on we will always assume that any depth-4 multilinear formula given to us is a simple formula.

## 2.3 ROABPs for Products of Sparse Polynomials

Another important model that we need for our constructions is that of Algebraic Branching Programs.

► **Definition 2.8** (Nisan [25]). An algebraic branching program (ABP) is a directed acyclic graph with one vertex  $s$  of in-degree zero (the *source*) and one vertex  $t$  of out-degree zero (the *sink*). The vertices of the graph are partitioned into levels labeled  $0, 1, \dots, D$ . Edges in the graph can only go from level  $\ell - 1$  to level  $\ell$ , for  $\ell \in [D]$ . The source is the only vertex at level 0 and the sink is the only vertex at level  $D$ . Each edge is labeled with an affine function in the input variables. The *width* of an ABP is the maximum number of nodes in any layer, and the *size* of an ABP is the number of vertices in the ABP.

Each path from  $s$  to  $t$  computes the polynomial which is the product of the labels of the path edges, and the ABP computes the sum, over all  $s \rightarrow t$  paths, of such polynomials.

► **Definition 2.9** (Ordered Read-Once Algebraic Branching Programs). A *Ordered Read-Once Algebraic Branching Program (ROABP)* in the variable set  $\{x_1, \dots, x_D\}$  is an ABP of depth  $D$ , such that each edge between layer  $\ell - 1$  and  $\ell$  is labeled by a univariate polynomial in  $x_\ell$ .

Below we show an elementary construction of ROABPs for a very specific class of polynomials, the proof of which can be found in [27].

► **Lemma 2.10.** *Let  $\mathbb{F}$  be a field, and  $f(y_1, \dots, y_m) = \sum_{i=1}^M \prod_{j=1}^{t_i} f_{i,j}$  be a multivariate polynomial over  $\mathbb{F}$ , such that for every  $1 \leq i \leq M$ :*

1. *At most  $k$  different  $1 \leq j \leq t_i$ , satisfy  $|\text{var}(f_{i,j})| > 1$ .*
2. *For every  $1 \leq j \leq t_i$ ,  $\|f_{i,j}\| \leq s$ .*

*Then  $f$  can be computed by an ROABP of width at most  $M \cdot s^k$ .*

## 2.4 Hashing

In this section we present the basic hashing tools that we will use in our construction. We first recall the notion of a  $k$ -wise independent hash family.

► **Definition 2.11.** A family of hash functions  $\mathcal{F} = \{h : [n] \rightarrow [m]\}$  is  $k$ -wise independent if for any  $k$  distinct elements  $(a_1, \dots, a_k) \in [n]^k$  and any  $k$  (not necessarily distinct) elements  $(b_1, \dots, b_k) \in [m]^k$ , we have:

$$\Pr_{h \in \mathcal{F}} [h(a_1) = b_1 \wedge \dots \wedge h(a_k) = b_k] = m^{-k}.$$

Our next lemma studies the case where several sets are hashed simultaneously by the same hash function. We present the lemma in a general form and only later, in the application, fix the parameters.

► **Lemma 2.12.** *Let  $0 < \delta < \epsilon$ , and  $n, M \in \mathbb{N}$  such that  $M = 2^{n^\delta}$ . Assume  $\mathcal{A}_1, \dots, \mathcal{A}_M$  are families of pairwise disjoint subsets of  $[n]$  such that for every  $1 \leq i \leq M$  and every  $A \in \mathcal{A}_i$ ,  $|A| \leq n^{1-\epsilon}$ . Let  $\gamma > 0$  be such that  $\gamma \geq (\epsilon - \delta)/2$ . Let  $\mathcal{F}$  be a family of  $k$ -wise independent hash functions from  $[n]$  to  $[m]$  for  $k = n^\delta + 2 \log n$  and  $m = 10n^{1-\epsilon+\gamma}$ .*

*Then there exists  $h \in \mathcal{F}$  such that for every  $1 \leq i \leq M$  and every  $1 \leq j \leq m$ , both of the following conditions hold:*

1. *For every set  $A \in \mathcal{A}_i$ ,  $|h^{-1}(j) \cap A| \leq k$ .*
2. *The number of sets  $A \in \mathcal{A}_i$  such that  $|h^{-1}(j) \cap A| > 1$  is at most  $k \log n$ .*

We conclude this section with the following well known fact (see, e.g., Chapter 16 in [7], and the references therein):

► **Fact 2.13.** *There exists an explicitly constructible family  $\mathcal{F}$  of  $k$ -wise independent hash functions from  $[n]$  to  $[10n^{1-\epsilon+\gamma}]$  of size  $|\mathcal{F}| = n^{O(k)}$ .*

### 3 Reducing the Bottom Variable Set of Depth-3 and Depth-4 Formulas

In this section we make the first step towards proving Theorems 1.1 and 1.3. As outlined in Section 1.4, our first step is making the functions computed at the bottom layers (linear functions in the case of depth-3 and “sparse” polynomials in the case of depth-4) have small variable set. Hence, we establish reductions from any  $\Sigma^{[M]}\Pi\Sigma$  or  $\Sigma^{[M]}\Pi\Sigma\Pi$  formula to a  $\Sigma^{[M]}\Pi\Sigma^{\{\tau\}}$  or  $\Sigma^{[M]}\Pi(\Sigma\Pi)^{\{\tau\}}$  formula, respectively. We first describe the simple depth-3 case. We then state without proofs the more general case of depth-4 formulas. The full proofs can be found in the full version ([27]).

#### 3.1 Reducing Bottom Variable Set for Depth-3

Given a depth-3 formula  $\sum_{i=1}^M \prod_{j=1}^{t_i} \ell_{i,j}$ , we would like to eliminate all linear functions that contain many variables. To this end, we observe that there must exist a variable that appears in many of these functions, and that taking a derivative according to that variable eliminates those functions from the formula.

► **Lemma 3.1.** *Let  $f(x_1, \dots, x_n) = \sum_{i=1}^M \prod_{j=1}^{t_i} \ell_{i,j}$  be a non-zero multilinear polynomial computed by a multilinear  $\Sigma^{[M]}\Pi\Sigma$  formula  $\Phi$  and let  $\epsilon > 0$ . Then, there exists a set of variables  $A$  of size  $|A| \leq \tilde{O}(n^\epsilon \cdot \log M)$  such that  $\partial_A f$  is a non-zero multilinear polynomial that can be computed by a multilinear  $\Sigma^{[M]}\Pi\Sigma^{\{n^{1-\epsilon}\}}$  formula.*

**Proof.** Define  $\mathcal{B} = \{\ell_{i,j} \mid |\text{var}(\ell_{i,j}) \cap \text{var}(f)| \geq n^{1-\epsilon}\}$  to be the set of “bad” linear functions. Those are linear functions that contain more than  $n^{1-\epsilon}$  variables that also appear in  $f$ . We show how to eliminate those linear functions from the formula while preserving non-zerosness.

Since for every  $\ell \in \mathcal{B}$ ,  $|\text{var}(\ell) \cap \text{var}(f)| \geq n^{1-\epsilon}$ , there exists a variable  $x_i$  that appears in at least  $|\mathcal{B}|n^{1-\epsilon}/n = |\mathcal{B}|/n^\epsilon$  linear functions in  $\mathcal{B}$  (and also in  $f$ ). The polynomial  $\partial_{x_i} f$  is non-zero, since  $x_i \in \text{var}(f)$ . Furthermore, using the fact that deriving with respect to a variable is a linear operation, and the fact that every multiplication gate in the formula multiplies linear functions with disjoint support, a formula for  $\partial_{x_i} f$  can be obtained from  $\Phi$  by replacing every linear function in which  $x_i$  appears with an appropriate constant. Therefore, every such function is removed from  $\mathcal{B}$ , and so the set of bad linear functions in  $\partial_{x_i} f$  is of size at most  $|\mathcal{B}| - |\mathcal{B}|/n^\epsilon = |\mathcal{B}| \cdot (1 - 1/n^\epsilon)$ . We continue this process for at most  $O(n^\epsilon \cdot \log |\mathcal{B}|)$  steps, until we reach a point where  $|\mathcal{B}| < 1$  and so  $|\mathcal{B}| = 0$ .

Finally, it remains to be noted that  $|\mathcal{B}| \leq Mn$ , since by multilinearity each multiplication gate multiplies linear functions with disjoint support, and so its fan-in is at most  $n$ . ◀

The process for depth-4 formulas follows the same outline as the depth-3 case, but there are a few more details, which can be found in the full version [27]. The precise statement of the support reduction for depth-4 is as follows:

► **Lemma 3.2** (Reduction to Depth-4 with Small Bottom Variable Set). *Let  $\Phi$  be a multilinear simple  $\Sigma^{[M]}\Pi\Sigma\Pi$  formula computing a non-zero multilinear polynomial  $f(x_1, \dots, x_n) \in \mathbb{F}[x_1, \dots, x_n]$ . There exist disjoint sets  $A, B \subset [n]$  with  $|A \sqcup B| \leq \frac{2n}{\tau} \cdot \log(|\Phi|)$  such that the polynomial  $\partial_A f|_{B=0}$  is non-zero and can be computed by a simple multilinear  $\Sigma^{[M]}\Pi(\Sigma\Pi)^{\{\tau\}}$  formula  $\Psi$ .*

#### 4 Hitting Set for $\Sigma\Pi\Sigma^{\{n^{1-\epsilon}\}}$ and $\Sigma\Pi(\Sigma\Pi)^{\{n^{1-\epsilon}\}}$ Formulas

In this section we construct subexponential sized hitting set for the classes of  $\Sigma^{[M]}\Pi\Sigma^{\{n^{1-\epsilon}\}}$  and  $\Sigma^{[M]}\Pi(\Sigma\Pi)^{\{n^{1-\epsilon}\}}$  multilinear formulas. Recall that in Section 3 we showed how to reduce general depth-3 and depth-4 formulas to these types of formulas. In the next section, we will show how to tie all loose edges and combine the arguments of Section 3 with those of this section in order to handle the general case.

An essential ingredient in our construction is a quasi-polynomial sized hitting set for Read-Once Algebraic Branching Programs (ROABPs) [14, 2]. We note that in our setting, we may assume that the reading order of the variables by the ABP is known.

► **Theorem 4.1** ([14, 2]). *Let  $\mathcal{C}$  be the class of  $n$ -variate polynomials computed by a ROABP of width  $w$ , such that the degree of each variable is at most  $d$ , over a field  $\mathbb{F}$  so that  $|\mathbb{F}| \geq \text{poly}(n, w, d)$ . Then  $\mathcal{C}$  has a hitting set of size  $\text{poly}(n, w, d)^{\log n}$  that can be constructed in time  $\text{poly}(n, w, d)^{\log n}$ .*

We begin by describing a unified construction for both  $\Sigma^{[M]}\Pi\Sigma^{\{n^{1-\epsilon}\}}$  and  $\Sigma^{[M]}\Pi(\Sigma\Pi)^{\{n^{1-\epsilon}\}}$  formulas. We then describe how to set the parameters of the construction for each of the cases.

► **Construction 4.2** (Hitting set for multilinear  $\Sigma^{[M]}\Pi\Sigma^{\{n^{1-\epsilon}\}}$  and  $\Sigma^{[M]}\Pi(\Sigma\Pi)^{\{n^{1-\epsilon}\}}$  formulas). *Let  $0 < \delta < \epsilon$  and  $n, k, s, M$  integers, such that  $M = 2^{n^\delta}$  and  $k = n^\delta + 2 \log n$ . Set  $m = 10n^{1-(\epsilon+\delta)/2}$  and  $t = k \log n$ . Let  $\mathcal{F}$  be a family of  $k$ -wise independent hash functions from  $[n]$  to  $[m]$ , as in Lemma 2.12. For every  $h \in \mathcal{F}$ , define the set  $I_h$  as follows:*

1. Partition the variables to sets<sup>4</sup>  $T_1 \sqcup T_2 \sqcup \dots \sqcup T_m = h^{-1}(1) \sqcup h^{-1}(2) \sqcup \dots \sqcup h^{-1}(m)$ .
2. For every  $1 \leq i \leq m$ , let  $\mathcal{H}_i$  be a hitting set for ROABPs of width  $M \cdot s^t$  and individual degree  $d = 1$  (as promised by Theorem 4.1), on the variables of  $T_i$  (of course,  $|T_i| \leq n$ ).
3. We define  $I_h$  as the set of all vectors  $v$  such that the restriction of  $v$  to the coordinates  $T_i$ ,  $v|_{T_i}$ , is in  $\mathcal{H}_i$ . I.e., in the notation of Section 2.1,

$$I_h = \mathcal{H}_1^{T_1} \times \mathcal{H}_2^{T_2} \times \dots \times \mathcal{H}_m^{T_m}.$$

Finally, define  $\mathcal{H} = \bigcup_{h \in \mathcal{F}} I_h$ .

The following lemma gives an upper bound to the size of the hitting set constructed in Construction 4.2.

<sup>4</sup> Recall that we associate subsets of  $[n]$  with subsets of the variables, and make no distinction in the notation.

► **Lemma 4.3.** *Let  $\delta, \epsilon, k, n, s$  and  $M$  be the parameters of Construction 4.2. The set  $\mathcal{H}$  constructed in Construction 4.2 has size  $n^{O(k)} \cdot (M \cdot s^{k \log n})^{\tilde{O}(n^{1-(\epsilon+\delta)/2})} = (M \cdot s^{k \log n})^{\tilde{O}(n^{1-(\epsilon+\delta)/2})}$ , and it can be constructed in time  $\text{poly}(|\mathcal{H}|)$ .*

**Proof.** This is a direct consequence of the construction, Fact 2.13 and Theorem 4.1. ◀

### 4.1 Depth-3 Formulas

We begin by describing the argument for depth-3 formulas. The following lemma proves that indeed, by setting the proper parameters, the set  $\mathcal{H}$  from Construction 4.2 does hit  $\Sigma^{[M]}\Pi\Sigma\{n^{1-\epsilon}\}$  formulas.

► **Lemma 4.4.** *Let  $f(x_1, \dots, x_n) \in \mathbb{F}[x_1, \dots, x_n]$  be a multilinear polynomial computed by a multilinear  $\Sigma^{[M]}\Pi\Sigma\{n^{1-\epsilon}\}$  formula  $\Phi = \sum_{i=1}^M \prod_{j=1}^{t_i} \ell_{i,j}$ . Let  $\mathcal{H}$  be the set constructed in Construction 4.2 with  $s = k + 1$ . Then there exists a point  $\bar{\alpha} \in \mathcal{H}$  such that  $f(\bar{\alpha}) \neq 0$ .*

**Proof.** For every multiplication gate  $1 \leq i \leq M$  in  $\Phi$ , define a partition of the variables

$$\mathcal{A}_i = \{\text{var}(\ell_{i,j}) \cap \text{var}(f) \mid 1 \leq j \leq t_i\}.$$

Let  $h \in \mathcal{F}$  be the function guaranteed by Lemma 2.12 with respect to the partitions  $\mathcal{A}_1, \dots, \mathcal{A}_M$ , and assume the setup of Construction 4.2. We claim that there exists  $\bar{\alpha} \in \mathcal{H}_h$  such that  $f(\bar{\alpha}) \neq 0$ .

To that end, consider the partition of the variables induced by  $h$ :

$$T_1 \sqcup T_2 \sqcup \dots \sqcup T_m = h^{-1}(1) \sqcup h^{-1}(2) \sqcup \dots \sqcup h^{-1}(m).$$

We view the polynomial as a polynomial  $f_1$  in the variables of  $T_1$ , over the extension field  $\mathbb{F}(T_2 \sqcup \dots \sqcup T_n)$ . We claim that  $f_1$  can be computed by an ROABP of width  $M \cdot (k + 1)^{k \log n}$ . To see this note that, by Lemma 2.12, in any multiplication gate, at most  $k \log n$  linear functions contain more than one variable from  $T_1$ , and each contains at most  $k$  variables. It follows that the sparsity of every linear function (with respect to the variables in  $T_1$ ) among those  $k \log n$  functions, is at most  $k + 1$ . By Lemma 2.10,  $f_1$  can be computed by an ROABP over  $\mathbb{F}(T_2 \sqcup \dots \sqcup T_n)$  of width  $M \cdot (k + 1)^{k \log n}$ . By the hitting set property of Theorem 4.1, there exists  $\bar{\alpha}_1 \in \mathcal{H}_1 \subseteq \mathbb{F}^{|T_1|}$  such that  $f_2 \stackrel{\text{def}}{=} f_1|_{T_1=\bar{\alpha}_1} \neq 0$ .

Similarly, the same conditions now hold for  $f_2$ , considered as a polynomial over the field  $\mathbb{F}(T_3 \sqcup \dots \sqcup T_n)$ , and so there exists  $\bar{\alpha}_2 \in \mathcal{H}_2 \subseteq \mathbb{F}^{|T_2|}$  such that  $f_3 \stackrel{\text{def}}{=} f_2|_{T_2=\bar{\alpha}_2} \neq 0$ .

We continue this process for  $m$  steps, and at the last step we find  $\bar{\alpha}_m$  such that  $f_{m-1}(\bar{\alpha}_m) = f(\bar{\alpha}_1, \dots, \bar{\alpha}_m) \neq 0$ , with  $(\bar{\alpha}_1, \dots, \bar{\alpha}_m) \in \mathbb{F}^n$  being the length  $n$  vector obtained by filling the entries of  $\bar{\alpha}_i \in \mathbb{F}^{|T_i|}$  in the positions indexed by  $T_i$ . ◀

### 4.2 Depth-4 Formulas

The argument for depth-4 formulas is very similar, apart from a small change in the setting of the parameters. Once again, the proof is omitted.

► **Lemma 4.5.** *Let  $f(x_1, \dots, x_n) \in \mathbb{F}[x_1, \dots, x_n]$  be a multilinear polynomial computed by a multilinear  $\Sigma^{[M]}\Pi(\Sigma\Pi)\{n^{1-\epsilon}\}$  formula  $\Phi = \sum_{i=1}^M \prod_{j=1}^{t_i} f_{i,j}$ . Let  $\mathcal{H}$  be the set constructed in Lemma 4.2 with  $s = 2^k$ . Then, there exists a point  $\bar{\alpha} \in \mathcal{H}$  such that  $f(\bar{\alpha}) \neq 0$ .*



## 5 Hitting Set for Depth-3 and Depth-4 Multilinear Formulas

Recall that, in Section 3, we showed that any non-zero  $\Sigma^{[M]}\Pi\Sigma$  or  $\Sigma^{[M]}\Pi\Sigma\Pi$  formula has a non-zero partial derivative (and, possibly, a restriction) which is computed by a non-zero  $\Sigma^{[M]}\Pi\Sigma^{\{n^{1-\epsilon}\}}$  or  $\Sigma^{[M]}\Pi(\Sigma\Pi)^{\{n^{1-\epsilon}\}}$  formula, respectively. Then, in Section 4 we gave hitting sets for such formulas. In this section we provide the final ingredient, which is to show how to “lift” those hitting sets back to the general class, via a simple method, albeit one that requires some notation.

Handling restrictions is fairly easy, and causes no blow up in the hitting set size: If we have a set  $\mathcal{H} \subseteq \mathbb{F}^{n-r}$  that hits  $f|_{B=0}$  for some multilinear polynomial  $f(x_1, \dots, x_n)$  and  $B \subseteq [n]$  with  $|B| = r$ , then simply extending  $\mathcal{H}$  into a subset of  $\mathbb{F}^n$  by filling out zeros in all the entries indexed by  $B$  will hit  $f$  itself.

In order to handle partial derivatives, first note that if  $f(x_1, \dots, x_n)$  is a multilinear polynomial, then

$$\partial_{x_i} f = f(x_1, \dots, x_{i-1}, 1, x_{i+1}, \dots, x_n) - f(x_1, \dots, x_{i-1}, 0, x_{i+1}, \dots, x_n),$$

and so if  $\partial_{x_i} f(\bar{\alpha}) \neq 0$  for some  $\bar{\alpha} \in \mathbb{F}^n$  then at least one of the two evaluations on the right hand side must be non-zero as well.

Applying this fact repeatedly, given a set  $A \subseteq [n]$  we can evaluate  $\partial_A f$  at any point by making  $2^{|A|}$  evaluations of  $f$ . Motivated by this, we introduce the following notation:

► **Definition 5.1.** Let  $f(x_1, \dots, x_n) \in \mathbb{F}[x_1, \dots, x_n]$  be a multilinear polynomial and  $A, B \subseteq [n]$  be two disjoint subsets of variables with  $|A| = r, |B| = r'$ . Let  $\mathcal{H} \subseteq \mathbb{F}^{n-(r+r')}$ .

We define the “lift” of  $\mathcal{H}$  with respect to  $(A, B)$  to be

$$\mathcal{L}_A^B(\mathcal{H}) = \left(\{0, 1\}^r\right)^A \times \left(\{0\}^{r'}\right)^B \times \mathcal{H}^{[n] \setminus (A \sqcup B)}.$$

In the special case where  $B = \emptyset$ , we simply denote  $\mathcal{L}_A^B(\mathcal{H}) = \mathcal{L}_A(\mathcal{H})$ .

That is, for all  $\bar{\alpha} \in \mathcal{H}$ ,  $\mathcal{L}_A^B(\mathcal{H})$  contains all the possible  $2^r$  ways to extend  $\bar{\alpha}$  into  $\bar{\beta} \in \mathbb{F}^n$  by filling out zeros and ones within the  $r$  entries that are indexed by  $A$ , and zeros in all the  $r'$  entries indexed by  $B$ .

### 5.1 Depth-3 Formulas

In this section we prove Theorem 1.1. For the reader’s convenience, we first restate the theorem:

► **Theorem 5.2** (Theorem 1.1, restated). *Let  $\mathcal{C}$  be the class of multilinear  $\Sigma^{[M]}\Pi\Sigma$  formulas for  $M = 2^{n^\delta}$ . There exists a hitting set  $\mathcal{H}$  of size  $|\mathcal{H}| = 2^{\tilde{O}(n^{2/3+2\delta/3})}$  for  $\mathcal{C}$ , that can be constructed in time  $\text{poly}(|\mathcal{H}|)$ .*

The size of the hitting set is subexponential for any constant  $\delta < 1/2$ . Also, if  $M = \text{poly}(n)$  then the size of the hitting set is  $2^{\tilde{O}(n^{2/3})}$ .

With Definition 5.1 in hand, we now provide our construction for  $\Sigma^{[M]}\Pi\Sigma$  formulas, towards the proof of Theorem 5.2.

► **Construction 5.3** (Hitting set for multilinear  $\Sigma^{[M]}\Pi\Sigma$  formulas). *Let  $M = 2^{n^\delta}$  and  $\epsilon = 2/3 - \delta/3$ . Let  $r = \tilde{O}(n^\epsilon \log M) = \tilde{O}(n^{\frac{2}{3} + \frac{2}{3}\delta})$  as guaranteed by Lemma 3.1. For every  $A \in \binom{[n]}{\leq r}$ ,*



construct a set  $\mathcal{H}_A \in \mathbb{F}^{n-|A|}$  using Construction 4.2 with parameters  $\delta, \epsilon, n, k, s = k + 1$  and  $M$  (recall that in Construction 5.3 we set  $k = n^\delta + 2 \log n$ ). Finally, let

$$\mathcal{H} = \bigcup_{A \in \binom{[n]}{\leq r}} \mathcal{L}_A(\mathcal{H}_A).$$

We are now ready to prove Theorem 5.2:

**Proof of Theorem 5.2.** We show that the set  $\mathcal{H}$  constructed in Construction 5.3 has the desired properties. First, note that by Lemma 4.3, for every  $A \subseteq [n]$  with

$$|A| \leq \tilde{O}(n^\epsilon \log M) = \tilde{O}(n^{2/3-\delta/3} \log M) = \tilde{O}(n^{\frac{2}{3}+\frac{2}{3}\delta}),$$

the set  $\mathcal{H}_A$  has size

$$(M \cdot (k + 1)^{k \log n})^{\tilde{O}(n^{2/3-\delta/3})} = 2^{\tilde{O}(n^{2/3+2\delta/3})},$$

where we have used the fact that, in Construction 5.3, we take  $k = n^\delta + 2 \log n$ . It therefore follows that

$$|\mathcal{L}_A(\mathcal{H}_A)| \leq 2^{|A|} \cdot |\mathcal{H}_A| = 2^{\tilde{O}(n^{2/3+2\delta/3})},$$

and that

$$|\mathcal{H}| \leq \sum_{i=0}^{\tilde{O}(n^{\frac{2}{3}+\frac{2}{3}\delta})} \sum_{A \subseteq [n], |A|=i} |\mathcal{L}_A(\mathcal{H}_A)| = 2^{\tilde{O}(n^{2/3+2\delta/3})}.$$

To show the hitting property of  $\mathcal{H}$ , let  $f(x_1, \dots, x_n)$  be a non-zero multilinear polynomial computed by a  $\Sigma^{[M]}\Pi\Sigma$  formula, and let  $A' \subseteq [n]$  be the set guaranteed by Lemma 3.1. Thus,  $|A'| \leq \tilde{O}(n^\epsilon \log M) = \tilde{O}(n^{\frac{2}{3}+\frac{2}{3}\delta})$ . Then by Lemma 4.4, there exists  $\bar{\alpha} \in \mathcal{H}_{A'}$  such that  $\partial_{A'} f(\bar{\alpha}) \neq 0$ , and so there must exist

$$\bar{\beta} \in \mathcal{L}_{A'}(\mathcal{H}_{A'}) \subseteq \mathcal{H}$$

such that  $f(\bar{\beta}) \neq 0$ . ◀

In the depth-4 case, the construction and proof are both very similar, with a slight change in the parameters. For the detailed statements, proofs and construction of the hitting set, we refer the reader to the full version [27].

## 6 Multilinear Depth- $d$ Regular Formulas

In this section, we consider *multilinear regular formulas*, which are regular formulas with the extra condition that each gate computes a multilinear polynomial. However, we will remove the bound on the formal degree of the formula. More precisely, we have the following definition:

► **Definition 6.1** (Multilinear Regular Formulas). We say that a formula  $\Phi$  is a multilinear  $(a_1, p_1, a_2, p_2, \dots, a_d, p_d, a_{d+1})$ -regular formula computing a multilinear polynomial  $f(x_1, \dots, x_n)$  if it can be computed by a multilinear  $\Sigma^{[a_1]}\Pi^{[p_1]}\Sigma^{[a_2]}\Pi^{[p_2]}\dots\Sigma^{[a_d]}\Pi^{[p_d]}\Sigma^{[a_{d+1}]}$ -formula. Notice that the size of such a formula is  $(\prod_{1 \leq i \leq d+1} a_i) \cdot (\prod_{1 \leq i \leq d} p_i)$  and the formal degree of such a formula is given by  $\deg(\Phi) = \prod_{1 \leq i \leq d} p_i$ . Since the formula is multilinear, we have that  $\deg(\Phi) \leq n$ .

Comparing with the definition given in Section 1.1, an  $(a_1, p_1, a_2, p_2, \dots, a_d, p_d, a_{d+1})$ -regular formula has depth  $2d + 1$ .

Our main result for multilinear regular formulas is given by the following theorem, a proof of which can be found in the full version of this paper [27].

► **Theorem 6.2** (Theorem 1.5, restated). *For  $d \geq 2$ , let  $\mathcal{C}_d$  be the class of multilinear polynomials computed by  $(a_1, p_1, a_2, p_2, \dots, a_d, p_d, a_{d+1})$ -regular formulas of size  $S \leq 2^{n^\delta}$  computing a multilinear polynomial  $f(x_1, \dots, x_n)$ , where  $\delta = \frac{1}{5^{d+1}}$ . Then, there exists a hitting set  $\mathcal{H}_d$  of size  $|\mathcal{H}_d| = 2^{\tilde{O}(n^{1-\delta/3})}$  for  $\mathcal{C}_d$ , that can be constructed in time  $\text{poly}(|\mathcal{H}_d|)$ .*

## 7 Lower Bounds for Bounded Depth Multilinear Formulas

As we noted earlier, the connection between construction of hitting sets and lower bounds for explicit polynomials is well established. The following theorem was proved by Heintz and Schnorr [16] and Agrawal [1], albeit we cite only a special case which matches our use of it:

► **Theorem 7.1** (A special case of [16, 1]). *Suppose there is a black-box deterministic PIT algorithm for a class  $\mathcal{C}$  of multilinear circuits, that outputs a hitting set  $\mathcal{H}$  of size  $|\mathcal{H}| = 2^{n^\alpha} < 2^n$  and runs in time  $\text{poly}(|\mathcal{H}|)$ , such that  $\mathcal{H}$  hits circuits of size at most  $2^{n^\delta}$ . Then, there exists a multilinear polynomial  $f(x_1, \dots, x_n)$  such that any circuit from the class  $\mathcal{C}$  computing  $f$  must be of size at least  $2^{n^\delta}$ , and the coefficients of  $f$  can be found in time  $2^{O(n)}$ .*

Theorem 7.1 is proved by finding a non-zero polynomial  $f(x_1, \dots, x_n)$  which vanishes on the entire hitting set  $\mathcal{H}$  of size  $2^{n^\alpha}$ , and then, by definition,  $f$  cannot have circuits of size  $2^{n^\delta}$ . Finding  $f$  amounts to finding a non-zero solution to a homogenous system of linear equations whose unknowns are the coefficients of the  $2^n$  possible multilinear monomials in  $x_1, \dots, x_n$ . As long as  $2^n > |\mathcal{H}| = 2^{n^\alpha}$ , a non-zero solution is guaranteed to exist.

Our lower bounds now follow as a direct application of our hitting set constructions and Theorem 7.1.

**Proofs of Corollaries 1.2, 1.4 and 1.6.** In light of Theorem 7.1, we only need to pick  $\delta$  so that the hitting sets we constructed have size less than  $2^n$ . The appropriate choices, by Theorems 1.1, 1.3 and 1.5, respectively, can be seen to be  $\delta = 1/2 - O(\log \log n / \log n)$  (for depth-3),  $\delta = 1/4 - O(\log \log n / \log n)$  (for depth-4) and  $\delta = \frac{1}{5^{\lfloor d/2 \rfloor + 1}} = O\left(\frac{1}{\sqrt{5}^d}\right)$  (for depth- $d$  regular formulas). ◀

## 8 Conclusion and Open Questions

We conclude this paper with some obvious open problems. First, as noted in Section 1.3, the lower bounds that we get from our hitting sets are not as good as the best lower bounds for these models. Can one improve our construction to yield lower bounds matching the best known lower bounds?

Currently, the size of the hitting set that we get for depth- $d$  regular multilinear formulas is roughly  $\exp(n^{1-1/\exp(d)})$ . Can the bound be improved to  $\exp(n^{1-\Omega(1/d)})$ ? Finally, another natural question is to extend our argument from depth- $d$  regular multilinear formulas to arbitrary depth- $d$  multilinear formulas.

**Acknowledgements.** The authors would like to thank Zeev Dvir and Avi Wigderson for helpful discussions during the course of this work.

## 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, R. Gurjar, A. Korwar, and N. Saxena. Hitting-sets for ROABP and sum of set-multilinear circuits. *Electronic Colloquium on Computational Complexity (ECCC)*, 21:85, 2014.
- 3 M. Agrawal, N. Kayal, and N. Saxena. Primes is in P. *Annals of Mathematics*, 160(2):781–793, 2004.
- 4 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 *STOC*, pages 599–614, 2012.
- 5 M. Agrawal, C. Saha, and N. Saxena. Quasi-polynomial hitting-set for set-depth- $\Delta$  formulas. In *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 321–330, 2013.
- 6 M. Agrawal and V. Vinay. Arithmetic circuits: A chasm at depth four. In *Proceedings of the 49th Annual FOCS*, pages 67–75, 2008.
- 7 N. Alon and J. H. Spencer. *The probabilistic method*. J. Wiley, 3 edition, 2008.
- 8 M. Anderson, D. van Melkebeek, and I. Volkovich. Derandomizing polynomial identity testing for multilinear constant-read formulae. In *Proceedings of the 26th Annual CCC*, pages 273–282, 2011.
- 9 R. A. DeMillo and R. J. Lipton. A probabilistic remark on algebraic program testing. *Inf. Process. Lett.*, 7(4):193–195, 1978.
- 10 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.
- 11 Z. Dvir, A. Shpilka, and A. Yehudayoff. Hardness-randomness tradeoffs for bounded depth arithmetic circuits. *SIAM J. on Computing*, 39(4):1279–1293, 2009.
- 12 M. A. Forbes, R. Saptharishi, and A. Shpilka. Hitting sets for multilinear read-once algebraic branching programs, in any order. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 – June 03, 2014*, pages 867–875, 2014.
- 13 M. A. Forbes and A. Shpilka. On identity testing of tensors, low-rank recovery and compressed sensing. In *Proceedings of the 44th annual STOC*, pages 163–172, 2012.
- 14 M. A. Forbes and A. Shpilka. Quasipolynomial-time identity testing of non-commutative and read-once oblivious algebraic branching programs. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 243–252, 2013.
- 15 A. Gupta, P. Kamath, N. Kayal, and R. Saptharishi. Arithmetic circuits: A chasm at depth three. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013*, pages 578–587, 2013.
- 16 J. Heintz and C. P. Schnorr. Testing polynomials which are easy to compute (extended abstract). In *Proceedings of the 12th annual STOC*, pages 262–272, 1980.
- 17 V. Kabanets and R. Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. In *Proceedings of the 35th Annual STOC*, pages 355–364, 2003.
- 18 V. Kabanets and R. Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Computational Complexity*, 13(1-2):1–46, 2004.
- 19 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. Comput.*, 42(6):2114–2131, 2013.
- 20 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.

- 21 N. Kayal, C. Saha, and R. Saptharishi. A super-polynomial lower bound for regular arithmetic formulas. In *Symposium on Theory of Computing, STOC 2014*, pages 146–153, 2014.
- 22 N. Kayal and S. Saraf. Blackbox polynomial identity testing for depth 3 circuits. In *Proceedings of the 50th Annual FOCS*, pages 198–207, 2009.
- 23 N. Kayal and N. Saxena. Polynomial identity testing for depth 3 circuits. *Computational Complexity*, 16(2):115–138, 2007.
- 24 P. Koiran. Arithmetic circuits: the chasm at depth four gets wider. *CoRR*, abs/1006.4700, 2010.
- 25 N. Nisan. Lower bounds for non-commutative computation. In *Proceedings of the 23rd Annual STOC*, pages 410–418, 1991.
- 26 N. Nisan and A. Wigderson. Lower bound on arithmetic circuits via partial derivatives. *Computational Complexity*, 6:217–234, 1996.
- 27 R. Oliveira, A. Shpilka, and B. L. Volk. Subexponential size hitting sets for bounded depth multilinear formulas. *Electronic Colloquium on Computational Complexity (ECCC)*, 21:157, 2014.
- 28 R. Raz and A. Shpilka. Deterministic polynomial identity testing in non-commutative models. *Computational Complexity*, 14(1):1–19, 2005.
- 29 R. Raz and A. Yehudayoff. Lower bounds and separations for constant depth multilinear circuits. *Computational Complexity*, 18(2):171–207, 2009.
- 30 S. Saraf and I. Volkovich. Black-box identity testing of depth-4 multilinear circuits. In *Proceedings of the 43rd annual STOC*, pages 421–430, 2011.
- 31 N. Saxena and C. Seshadhri. Blackbox identity testing for bounded top fanin depth-3 circuits: the field doesn’t matter. In *Proceedings of the 43rd Annual STOC*, pages 431–440, 2011.
- 32 J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *J. ACM*, 27(4):701–717, 1980.
- 33 A. Shpilka and I. Volkovich. Read-once polynomial identity testing. In *Proceedings of the 40th Annual STOC*, pages 507–516, 2008.
- 34 A. Shpilka and I. Volkovich. Improved polynomial identity testing for read-once formulas. In *APPROX-RANDOM*, pages 700–713, 2009.
- 35 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.
- 36 S. Tavenas. Improved bounds for reduction to depth 4 and depth 3. In *Mathematical Foundations of Computer Science 2013 – 38th International Symposium, MFCS 2013*, pages 813–824, 2013.
- 37 L. G. Valiant, S. Skyum, S. Berkowitz, and C. Rackoff. Fast parallel computation of polynomials using few processors. *SIAM J. on Computing*, 12(4):641–644, November 1983.
- 38 R. Zippel. Probabilistic algorithms for sparse polynomials. In *EUROSAM*, pages 216–226, 1979.

# Deterministic Identity Testing for Sum of Read-once Oblivious Arithmetic Branching Programs

Rohit Gurjar<sup>\*1</sup>, Arpita Korwar<sup>1</sup>, Nitin Saxena<sup>†1</sup>, and Thomas Thierauf<sup>‡2</sup>

- 1 Department of Computer Science and Engineering, IIT Kanpur, India  
{rgurjar, arpk, nitin}@iitk.ac.in
- 2 Aalen University, Germany  
thomas.thierauf@htw-aalen.de

---

## Abstract

A *read-once oblivious arithmetic branching program (ROABP)* is an arithmetic branching program (ABP) where each variable occurs in at most one layer. We give the first polynomial time whitebox identity test for a polynomial computed by a sum of constantly many ROABPs. We also give a corresponding blackbox algorithm with quasi-polynomial time complexity  $n^{O(\log n)}$ . In both the cases, our time complexity is double exponential in the number of ROABPs.

ROABPs are a generalization of set-multilinear depth-3 circuits. The prior results for the sum of constantly many set-multilinear depth-3 circuits were only slightly better than brute-force, i.e. exponential-time.

Our techniques are a new interplay of three concepts for ROABP: low evaluation dimension, basis isolating weight assignment and low-support rank concentration. We relate basis isolation to rank concentration and extend it to a sum of two ROABPs using evaluation dimension (or partial derivatives).

**1998 ACM Subject Classification** F.1.3 Complexity Measures and Classes, F.2.1 Numerical Algorithms and Problems

**Keywords and phrases** PIT, hitting-set, Sum of ROABPs, Evaluation Dimension, Rank Concentration

**Digital Object Identifier** 10.4230/LIPIcs.CCC.2015.323

## 1 Introduction

Polynomial Identity Testing (PIT) is the problem of testing whether a given  $n$ -variate polynomial is identically zero or not. The input to the PIT problem may be in the form of arithmetic circuits or arithmetic branching programs (ABP). They are the arithmetic analogues of boolean circuits and boolean branching programs, respectively. It is well known that PIT can be solved in randomized polynomial time, see e.g. [29]. The randomized algorithm just evaluates the polynomial at random points; thus, it is a *blackbox* algorithm. In contrast, an algorithm is a *whitebox* algorithm if it looks inside the given circuit or branching program. We consider both, whitebox and blackbox algorithms.

---

\* Supported by TCS Ph. D research fellowship.

† Supported by DST-SERB.

‡ Supported by DFG grant TH 472/4-1.



Since all problems with randomized polynomial-time solutions are conjectured to have deterministic polynomial-time algorithms, we expect that such an algorithm exists for PIT. It is also known that any sub-exponential time algorithm for PIT implies a lower bound [15, 2]. See also the surveys [25, 26, 30].

An efficient deterministic solution for PIT is known only for very restricted input models, for example, sparse polynomials [5, 19], constant fan-in depth-3 ( $\Sigma\Pi\Sigma$ ) circuits [7, 18, 17, 16, 27, 28], set-multilinear circuits [22, 10, 4], read-once oblivious ABP (ROABP) [22, 12, 9, 3]. This lack of progress is not surprising: Gupta et al. [13] showed that a polynomial time test for depth-3 circuits would imply a sub-exponential time test for general circuits. For now, even a sub-exponential solution for depth-3 circuits seems elusive. However, an efficient test for depth-3 multilinear circuits looks within reach as a lower bound against this class of circuits is already known [23]. A circuit is called *multilinear* if all its gates compute a multilinear polynomial, i.e. polynomials such that the maximum degree of any variable is one.

A depth-3 multilinear circuit is called *set-multilinear* if all the product gates in it induce the same partition on the set of variables. It is easy to see that a depth-3 multilinear circuit is a sum of polynomially many set-multilinear circuits. Hence, a natural first step to attack depth-3 multilinear circuit is to find an efficient test for the sum of two set-multilinear polynomials. Before this work, the only non-trivial test known for sum of two set-multilinear circuits was a sub-exponential whitebox algorithm by Agrawal et al. [3]. Subsequently, a sub-exponential time blackbox test was also given for depth-3 multilinear circuits [6]. Our results imply the first polynomial-time whitebox algorithm, and the first quasi-polynomial-time blackbox algorithm, for the sum of two set-multilinear circuits.

In this paper, we deal with ROABPs, a model which subsumes set-multilinear circuits; see for example [3, Lemma 14]. A read-once oblivious ABP (ROABP) is an arithmetic branching program, where each variable occurs in at most one layer. There has been a long chain of work on identity testing for ROABP, see the thesis of Michael Forbes [8] for an excellent overview. In 2005, Raz and Shpilka [22] gave a polynomial-time whitebox test for ROABP. Then, Forbes and Shpilka [12] gave an  $s^{O(\log n)}$ -time blackbox algorithm for ROABP with known variable order, where  $s$  is the size of the ROABP and  $n$  is number of variables. This was followed by a complete blackbox test [9] that took  $s^{O(d \log^2 s)}$  steps, where  $d$  is the syntactic degree bound of any variable. This was further improved by Agrawal et al. [3] to  $s^{O(\log n)}$  time. They removed the exponential dependence on the degree  $d$ . Their test is based on the idea of *basis isolating weight assignment*. Given a polynomial over an algebra, it assigns weights to the variables, and naturally extends it to monomials, such that there is a unique minimum weight basis among the coefficients of the polynomial.

In another work, Jansen et al. [14] gave a blackbox test for a sum of constantly many “ROABPs”. Their definition of “ROABP” is much weaker. They assume that a variable appears on at most one edge in the ABP.

We consider the sum of ROABPs. Note that there are polynomials  $P(\mathbf{x})$  computed by the sum of two ROABPs such that any single ROABP that computes  $P(\mathbf{x})$  has exponential size [20]. Hence, the previous results on single ROABPs do not help here. In Section 3 we show our first main result (Theorem 3.2):

*PIT for the sum of constantly many ROABPs is in polynomial time.*

The exact time bound we get for the PIT-algorithm is  $(ndw^{2^c})^{O(c)}$ , where  $n$  is the number of variables,  $d$  is the degree bound of the variables,  $c$  is the number of ROABPs and  $w$  is their width. Hence our time bound is double exponential in  $c$ , but polynomial in  $ndw$ .

Our algorithm uses the fact that the *evaluation dimension* of an ROABP is equal to the width of the ROABP [21, 11]. Namely, we consider a set of linear dependencies derived from partial evaluations of the ROABPs<sup>1</sup>. We view identity testing of the sum of two ROABPs as testing the equivalence of two ROABPs. Our idea is inspired from a similar result in the boolean case. Testing the equivalence of two ordered boolean branching programs (OBDD) is in polynomial time [24]. OBDDs too have a similar property of small evaluation dimension, except that the notion of *linear dependence* becomes *equality* in the boolean setting. Our equivalence test, for two ROABPs  $A$  and  $B$ , takes linear dependencies among partial evaluations of  $A$  and verifies them for the corresponding partial evaluations of  $B$ . As  $B$  is an ROABP, the verification of these dependencies reduces to identity testing for a single ROABP.

In Section 3.2, we generalize this test to the sum of  $c$  ROABPs. There we take  $A$  as one ROABP and  $B$  as the sum of the remaining  $c - 1$  ROABPs. In this case, the verification of the dependencies for  $B$  becomes the question of identity testing of a sum of  $c - 1$  ROABPs, which we solve recursively.

The same idea can be applied to decide the equivalence of an OBDD with the XOR of  $c - 1$  OBDDs. We skip these details here<sup>1</sup> as we are mainly interested in the arithmetic case.

In Section 4, we give an identity test for a sum of ROABPs in the blackbox setting. That is, we are given blackbox access to a sum of ROABPs and *not* to the individual ROABPs. Our main result here is as follows (Theorem 4.9):

*There is a blackbox PIT for the sum of constantly many ROABPs that works in quasi-polynomial time.*

The exact time bound we get for the PIT-algorithm is  $(ndw)^{O(c2^c \log(ndw))}$ , where  $n$  is the number of variables,  $d$  is the degree bound of the variables,  $c$  is the number of ROABPs and  $w$  is their width. Hence our time bound is double exponential in  $c$ , and quasi-polynomial in  $n, d, w$ .

Here again, using the low evaluation dimension property, the question is reduced to identity testing for a single ROABP. But, just a hitting-set for ROABP does not suffice here, we need an efficient shift of the variables which gives low-support concentration in any polynomial computed by an ROABP. An  $\ell$ -concentration in a polynomial  $P(\mathbf{x})$  means that all of its coefficients are in the linear span of its coefficients corresponding to monomials with support  $< \ell$ . Essentially we show that a shift, which achieves low-support concentration for an ROABP of width  $w^{2^c}$ , also works for a sum of  $c$  ROABPs (Lemma 4.8). This is surprising, because as mentioned above, a sum of  $c$  ROABPs is not captured by an ROABP with polynomially bounded width [20].

A novel part of our proof is the idea that for a polynomial over a  $k$ -dimensional  $\mathbb{F}$ -algebra  $\mathbb{A}_k$ , a shift by a basis isolating weight assignment achieves low-support concentration. To elaborate, let  $w: \mathbf{x} \rightarrow \mathbb{N}$  be a basis isolating weight assignment for a polynomial  $P(\mathbf{x}) \in \mathbb{A}_k[\mathbf{x}]$  then  $P(\mathbf{x} + t^w)$  has  $O(\log k)$ -concentration over  $\mathbb{F}(t)$ . As Agrawal et al. [3] gave a basis isolating weight assignment for ROABPs, we can use it to get low-support concentration. Forbes et al. [9] had also achieved low-support concentration in ROABPs, but with a higher cost. Our concentration proof significantly differs from the older rank concentration proofs [4, 9], which always assume *distinct* weights for all the monomials or coefficients.

<sup>1</sup> Equivalently, we work with the dependencies of the partial derivatives.



Here, we only require that the weight of a coefficient is greater than the weight of the basis coefficients that it depends on.

## 2 Preliminaries

### 2.1 Notation

Let  $\mathbf{x} = (x_1, x_2, \dots, x_n)$  be a tuple of  $n$  variables. For any  $\mathbf{a} = (a_1, a_2, \dots, a_n) \in \mathbb{N}^n$ , we denote by  $\mathbf{x}^{\mathbf{a}}$  the monomial  $\prod_{i=1}^n x_i^{a_i}$ . The *support size* of a monomial  $\mathbf{x}^{\mathbf{a}}$  is given by  $\text{supp}(\mathbf{a}) = |\{a_i \neq 0 \mid i \in [n]\}|$ .

Let  $\mathbb{F}$  be some field. Let  $A(\mathbf{x})$  be a polynomial over  $\mathbb{F}$  in  $n$  variables. A polynomial  $A(\mathbf{x})$  is said to have individual degree  $d$ , if the degree of each variable is bounded by  $d$  for each monomial in  $A(\mathbf{x})$ . When  $A(\mathbf{x})$  has individual degree  $d$ , then the exponent  $\mathbf{a}$  of any monomial  $\mathbf{x}^{\mathbf{a}}$  of  $A(\mathbf{x})$  is in the set

$$M = \{0, 1, \dots, d\}^n.$$

By  $\text{coeff}_A(\mathbf{x}^{\mathbf{a}}) \in \mathbb{F}$  we denote the coefficient of the monomial  $\mathbf{x}^{\mathbf{a}}$  in  $A(\mathbf{x})$ . Hence, we can write

$$A(\mathbf{x}) = \sum_{\mathbf{a} \in M} \text{coeff}_A(\mathbf{x}^{\mathbf{a}}) \mathbf{x}^{\mathbf{a}}.$$

The *sparsity* of polynomial  $A(\mathbf{x})$  is the number of nonzero coefficients  $\text{coeff}_A(\mathbf{x}^{\mathbf{a}})$ .

We also consider *matrix polynomials* where the coefficients  $\text{coeff}_A(\mathbf{x}^{\mathbf{a}})$  are  $w \times w$  matrices, for some  $w$ . In an abstract setting, these are polynomials over a  $w^2$ -dimensional  $\mathbb{F}$ -algebra  $\mathbb{A}$ . Recall that an  $\mathbb{F}$ -algebra is a vector space over  $\mathbb{F}$  with a multiplication which is bilinear and associative, i.e.  $\mathbb{A}$  is a ring. The *coefficient space* is then defined as the span of all coefficients of  $A$ , i.e.,  $\text{span}_{\mathbb{F}}\{\text{coeff}_A(\mathbf{x}^{\mathbf{a}}) \mid \mathbf{a} \in M\}$ .

Consider a partition of the variables  $\mathbf{x}$  into two parts  $\mathbf{y}$  and  $\mathbf{z}$ , with  $|\mathbf{y}| = k$ . A polynomial  $A(\mathbf{x})$  can be viewed as a polynomial in variables  $\mathbf{y}$ , where the coefficients are polynomials in  $\mathbb{F}[\mathbf{z}]$ . For monomial  $\mathbf{y}^{\mathbf{a}}$ , let us denote the coefficient of  $\mathbf{y}^{\mathbf{a}}$  in  $A(\mathbf{x})$  by  $A_{(\mathbf{y}, \mathbf{a})} \in \mathbb{F}[\mathbf{z}]$ . For example, in the polynomial  $A(\mathbf{x}) = x_1 + x_1x_2$ , we have  $A_{(\{x_1\}, 1)} = 1 + x_2$ , whereas  $\text{coeff}_A(x_1) = 1$ .

Thus,  $A(\mathbf{x})$  can be written as

$$A(\mathbf{x}) = \sum_{\mathbf{a} \in \{0, 1, \dots, d\}^k} A_{(\mathbf{y}, \mathbf{a})} \mathbf{y}^{\mathbf{a}}. \tag{1}$$

The coefficient  $A_{(\mathbf{y}, \mathbf{a})}$  is also sometimes expressed in the literature as a partial derivative  $\frac{\partial A}{\partial \mathbf{y}^{\mathbf{a}}}$  evaluated at  $\mathbf{y} = \mathbf{0}$  (and multiplied by an appropriate constant), see [11, Section 6].

For a set of polynomials  $\mathcal{P}$ , we define their  $\mathbb{F}$ -span as

$$\text{span}_{\mathbb{F}} \mathcal{P} = \left\{ \sum_{A \in \mathcal{P}} \alpha_A A \mid \alpha_A \in \mathbb{F} \text{ for all } A \in \mathcal{P} \right\}.$$

The set of polynomials  $\mathcal{P}$  is said to be  $\mathbb{F}$ -linearly independent if  $\sum_{A \in \mathcal{P}} \alpha_A A = 0$  holds only for  $\alpha_A = 0$ , for all  $A \in \mathcal{P}$ . The *dimension*  $\text{dim}_{\mathbb{F}} \mathcal{P}$  of  $\mathcal{P}$  is the cardinality of the largest  $\mathbb{F}$ -linearly independent subset of  $\mathcal{P}$ .

For a matrix  $R$ , we denote by  $R(i, \cdot)$  and  $R(\cdot, i)$  the  $i$ -th row and the  $i$ -th column of  $R$ , respectively. For any  $a \in \mathbb{F}^{k \times k'}$ ,  $b \in \mathbb{F}^{\ell \times \ell'}$ , the tensor product of  $a$  and  $b$  is denoted by  $a \otimes b$ . The inner product is denoted by  $\langle a, b \rangle$ . We abuse this notation slightly: for any  $a, R \in \mathbb{F}^{w \times w}$ , let  $\langle a, R \rangle = \sum_{i=1}^w \sum_{j=1}^w a_{ij} R_{ij}$ .



## 2.2 Arithmetic branching programs

An *arithmetic branching program* (ABP) is a directed graph with  $\ell + 1$  layers of vertices  $(V_0, V_1, \dots, V_\ell)$ . The layers  $V_0$  and  $V_\ell$  each contain only one vertex, the *start node*  $v_0$  and the *end node*  $v_\ell$ , respectively. The edges are only going from the vertices in the layer  $V_{i-1}$  to the vertices in the layer  $V_i$ , for any  $i \in [d]$ . All the edges in the graph have weights from  $\mathbb{F}[\mathbf{x}]$ , for some field  $\mathbb{F}$ . The *length* of an ABP is the length of a longest path in the ABP, i.e.  $\ell$ . An ABP has *width*  $w$ , if  $|V_i| \leq w$  for all  $1 \leq i \leq \ell - 1$ .

For an edge  $e$ , let us denote its weight by  $W(e)$ . For a path  $p$ , its weight  $W(p)$  is defined to be the product of weights of all the edges in it,

$$W(p) = \prod_{e \in p} W(e).$$

The *polynomial*  $A(\mathbf{x})$  computed by the ABP is the sum of the weights of all the paths from  $v_0$  to  $v_\ell$ ,

$$A(\mathbf{x}) = \sum_{p \text{ path } v_0 \rightsquigarrow v_\ell} W(p).$$

Let the set of nodes in  $V_i$  be  $\{v_{i,j} \mid j \in [w]\}$ . The branching program can alternately be represented by a matrix product  $\prod_{i=1}^{\ell} D_i$ , where  $D_1 \in \mathbb{F}[\mathbf{x}]^{1 \times w}$ ,  $D_i \in \mathbb{F}[\mathbf{x}]^{w \times w}$  for  $2 \leq i \leq \ell - 1$ , and  $D_\ell \in \mathbb{F}[\mathbf{x}]^{w \times 1}$  such that

$$\begin{aligned} D_1(j) &= W(v_0, v_{1,j}), \text{ for } 1 \leq j \leq w, \\ D_i(j, k) &= W(v_{i-1,j}, v_{i,k}), \text{ for } 1 \leq j, k \leq w \text{ and } 2 \leq i \leq \ell - 1, \\ D_\ell(k) &= W(v_{\ell-1,k}, v_\ell), \text{ for } 1 \leq k \leq w. \end{aligned}$$

Here we use the convention that  $W(u, v) = 0$  if  $(u, v)$  is not an edge in the ABP.

## 2.3 Read-once oblivious arithmetic branching programs

An ABP is called a *read-once oblivious ABP* (ROABP) if the edge weights in every layer are univariate polynomials in the same variable, and every variable occurs in at most one layer. Hence, the length of an ROABP is  $n$ , the number of variables. The entries in the matrix  $D_i$  defined above come from  $\mathbb{F}[x_{\pi(i)}]$ , for all  $i \in [n]$ , where  $\pi$  is a permutation on the set  $[n]$ . The order  $(x_{\pi(1)}, x_{\pi(2)}, \dots, x_{\pi(n)})$  is said to be the *variable order* of the ROABP.

We will view  $D_i$  as a polynomial in the variable  $x_{\pi(i)}$ , whose coefficients are  $w$ -dimensional vectors or matrices. Namely, for an exponent  $\mathbf{a} = (a_1, a_2, \dots, a_n)$ , the coefficient of

- $x_{\pi(1)}^{a_{\pi(1)}}$  in  $D_1(x_{\pi(1)})$  is the row vector  $\text{coeff}_{D_1}(x_{\pi(1)}^{a_{\pi(1)}}) \in \mathbb{F}^{1 \times w}$ ,
- $x_{\pi(i)}^{a_{\pi(i)}}$  in  $D_i(x_{\pi(i)})$  is the matrix  $\text{coeff}_{D_i}(x_{\pi(i)}^{a_{\pi(i)}}) \in \mathbb{F}^{w \times w}$ , for  $i = 2, 3, \dots, n - 1$ , and
- $x_{\pi(n)}^{a_{\pi(n)}}$  in  $D_n(x_{\pi(n)})$  is the vector  $\text{coeff}_{D_n}(x_{\pi(n)}^{a_{\pi(n)}}) \in \mathbb{F}^{w \times 1}$ .

The read once property gives us an easy way to express the coefficients of the polynomial  $A(\mathbf{x})$  computed by an ROABP.

► **Lemma 2.1.** *For a polynomial  $A(\mathbf{x}) = D_1(x_{\pi(1)})D_2(x_{\pi(2)}) \cdots D_n(x_{\pi(n)})$  computed by an ROABP, we have*

$$\text{coeff}_A(\mathbf{x}^{\mathbf{a}}) = \prod_{i=1}^n \text{coeff}_{D_i}(x_{\pi(i)}^{a_{\pi(i)}}) \in \mathbb{F}. \tag{2}$$

We also consider matrix polynomials computed by an ROABP. A matrix polynomial  $A(\mathbf{x}) \in F^{w \times w}[\mathbf{x}]$  is said to be computed by an ROABP if  $A = D_1 D_2 \cdots D_n$ , where  $D_i \in F^{w \times w}[x_{\pi(i)}]$  for  $i = 1, 2, \dots, n$  and some permutation  $\pi$  on  $[n]$ . Similarly, a vector polynomial  $A(\mathbf{x}) \in F^{1 \times w}[\mathbf{x}]$  is said to be computed by an ROABP if  $A = D_1 D_2 \cdots D_n$ , where  $D_1 \in F^{1 \times w}[x_{\pi(1)}]$  and  $D_i \in F^{w \times w}[x_{\pi(i)}]$  for  $i = 2, \dots, n$ . Usually, we will assume that an ROABP computes a polynomial in  $\mathbb{F}[\mathbf{x}]$ , unless mentioned otherwise.

Let  $A(\mathbf{x})$  be the polynomial computed by an ROABP and let  $\mathbf{y}$  and  $\mathbf{z}$  be a partition of the variables  $\mathbf{x}$  such that  $\mathbf{y}$  is a *prefix* of the variable order of the ROABP. Recall from equation (1) that  $A_{(\mathbf{y}, \mathbf{a})} \in \mathbb{F}[\mathbf{z}]$  is the coefficient of monomial  $\mathbf{y}^{\mathbf{a}}$  in  $A(\mathbf{x})$ . Nisan [21] showed that for every prefix  $\mathbf{y}$ , the dimension of the set of coefficient polynomials  $A_{(\mathbf{y}, \mathbf{a})}$  is bounded by the width of the ROABP<sup>2</sup>. This holds in spite of the fact that the number of these polynomials is large.

► **Lemma 2.2** ([21], Prefix  $\mathbf{y}$ ). *Let  $A(\mathbf{x})$  be a polynomial of individual degree  $d$ , computed by an ROABP of width  $w$  with variable order  $(x_1, x_2, \dots, x_n)$ . Let  $k \leq n$  and  $\mathbf{y} = (x_1, x_2, \dots, x_k)$  be the prefix of length  $k$  of  $\mathbf{x}$ . Then  $\dim_{\mathbb{F}}\{A_{(\mathbf{y}, \mathbf{a})} \mid \mathbf{a} \in \{0, 1, \dots, d\}^k\} \leq w$ .*

**Proof.** Let  $A(\mathbf{x}) = D_1(x_1) D_2(x_2) \cdots D_n(x_n)$ , where  $D_1 \in \mathbb{F}^{1 \times w}[x_1]$ ,  $D_n \in \mathbb{F}^{w \times 1}[x_n]$  and  $D_i \in \mathbb{F}^{w \times w}[x_i]$ , for  $2 \leq i \leq n-1$ . Let  $\mathbf{z} = (x_{k+1}, x_{k+2}, \dots, x_n)$  be the remaining variables of  $\mathbf{x}$ . Define  $P(\mathbf{y}) = D_1 D_2 \cdots D_k$  and  $Q(\mathbf{z}) = D_{k+1} D_{k+2} \cdots D_n$ . Then  $P$  and  $Q$  are vectors of length  $w$ ,

$$\begin{aligned} P(\mathbf{y}) &= [P_1(\mathbf{y}) \ P_2(\mathbf{y}) \ \cdots \ P_w(\mathbf{y})] \\ Q(\mathbf{z}) &= [Q_1(\mathbf{z}) \ Q_2(\mathbf{z}) \ \cdots \ Q_w(\mathbf{z})]^T \end{aligned}$$

where  $P_i(\mathbf{y}) \in \mathbb{F}[\mathbf{y}]$  and  $Q_i(\mathbf{z}) \in \mathbb{F}[\mathbf{z}]$ , for  $1 \leq i \leq w$ , and we have  $A(\mathbf{x}) = P(\mathbf{y}) Q(\mathbf{z})$ .

We get the following generalization of equation (2): for any  $\mathbf{a} \in \{0, 1, \dots, d\}^k$ , the coefficient  $A_{(\mathbf{y}, \mathbf{a})} \in \mathbb{F}[\mathbf{z}]$  of monomial  $\mathbf{y}^{\mathbf{a}}$  can be written as

$$A_{(\mathbf{y}, \mathbf{a})} = \sum_{i=1}^w \text{coeff}_{P_i}(\mathbf{y}^{\mathbf{a}}) Q_i(\mathbf{z}). \quad (3)$$

That is, every  $A_{(\mathbf{y}, \mathbf{a})}$  is in the  $\mathbb{F}$ -span of the polynomials  $Q_1, Q_2, \dots, Q_w$ . Hence, the claim follows. ◀

Observe that equation (3) tells us that the polynomials  $A_{(\mathbf{y}, \mathbf{a})}$  can also be computed by an ROABP of width  $w$ : by equation (2), we have  $\text{coeff}_{P_i}(\mathbf{y}^{\mathbf{a}}) = \prod_{x_i \in \mathbf{y}} \text{coeff}_{D_i}(x_i^{a_i})$ . Hence, in the ROABP for  $A$  we simply have to replace the matrices  $D_i$  which belong to  $P$  by the coefficient matrices  $\text{coeff}_{D_i}(x_i^{a_i})$ . Here,  $\mathbf{y}$  is a prefix of  $\mathbf{x}$ . But this is not necessary for the construction to work. The variables in  $\mathbf{y}$  can be arbitrarily distributed in  $\mathbf{x}$ . We summarize the observation in the following lemma.

► **Lemma 2.3** (Arbitrary  $\mathbf{y}$ ). *Let  $A(\mathbf{x})$  be a polynomial of individual degree  $d$ , computed by an ROABP of width  $w$  and  $\mathbf{y} = (x_{i_1}, x_{i_2}, \dots, x_{i_k})$  be any  $k$  variables of  $\mathbf{x}$ . Then the polynomial  $A_{(\mathbf{y}, \mathbf{a})}$  can be computed by an ROABP of width  $w$ , for every  $\mathbf{a} \in \{0, 1, \dots, d\}^k$ . Moreover, all these ROABPs have the same variable order, inherited from the order of the ROABP for  $A$ .*

For a general polynomial, the dimension considered in Lemma 2.2 can be exponentially large in  $n$ . We will next show the converse of Lemma 2.2: if this dimension is small for a

<sup>2</sup> Nisan [21] showed it for non-commutative ABP, but the same proof works for ROABP.

polynomial then there exists a small width ROABP for that polynomial. Hence, this property characterizes the class of polynomials computed by ROABPs. Forbes et al. [11, Section 6] give a similar characterization in terms of evaluation dimension, for polynomials which can be computed by an ROABP, in any variable order. On the other hand, we work with a fixed variable order.

As a preparation to prove this characterization we define a *characterizing set of dependencies* of a polynomial  $A(\mathbf{x})$  of individual degree  $d$ , with respect to a variable order  $(x_1, x_2, \dots, x_n)$ . This set of dependencies will essentially give us an ROABP for  $A$  in the variable order  $(x_1, x_2, \dots, x_n)$ .

► **Definition 2.4.** Let  $A(\mathbf{x})$  be polynomial of individual degree  $d$ , where  $\mathbf{x} = (x_1, x_2, \dots, x_n)$ . For any  $0 \leq k \leq n$  and  $\mathbf{y}_k = (x_1, x_2, \dots, x_k)$ , let

$$\dim_{\mathbb{F}}\{A_{(\mathbf{y}_k, \mathbf{a})} \mid \mathbf{a} \in \{0, 1, \dots, d\}^k\} \leq w,$$

for some  $w$ .

For  $0 \leq k \leq n$ , we define the *spanning sets*  $\text{span}_k(A)$  and the *dependency sets*  $\text{depend}_k(A)$  as subsets of  $\{0, 1, \dots, d\}^k$  as follows.

For  $k = 0$ , let  $\text{depend}_0(A) = \emptyset$  and  $\text{span}_0(A) = \{\epsilon\}$ , where  $\epsilon = ()$  denotes the empty tuple. For  $k > 0$ , let

- $\text{depend}_k(A) = \{(\mathbf{a}, j) \mid \mathbf{a} \in \text{span}_{k-1}(A) \text{ and } 0 \leq j \leq d\}$ , i.e.  $\text{depend}_k(A)$  contains all possible extensions of the tuples in  $\text{span}_{k-1}(A)$ .
- $\text{span}_k(A) \subseteq \text{depend}_k(A)$  is any set of size  $\leq w$ , such that for any  $\mathbf{b} \in \text{depend}_k(A)$ , the polynomial  $A_{(\mathbf{y}_k, \mathbf{b})}$  is in the span of  $\{A_{(\mathbf{y}_k, \mathbf{a})} \mid \mathbf{a} \in \text{span}_k(A)\}$ .

The dependencies of the polynomials in  $\{A_{(\mathbf{y}_k, \mathbf{a})} \mid \mathbf{a} \in \text{depend}_k(A)\}$  over  $\{A_{(\mathbf{y}_k, \mathbf{a})} \mid \mathbf{a} \in \text{span}_k(A)\}$  are the *characterizing set of dependencies*.

The definition of  $\text{span}_k(A)$  is not unique. For our purpose, it does not matter which of the possibilities we take, we simply fix one of them. We do *not* require that  $\text{span}_k(A)$  is of minimal size, i.e. the polynomials associated with  $\text{span}_k(A)$  constitute a basis for the polynomials associated with  $\text{depend}_k(A)$ . This is because in the whitebox test in Section 3, we will efficiently construct the sets  $\text{span}_k(A)$ , and there we cannot guarantee to obtain a basis. We will see that it suffices to have  $|\text{span}_k(A)| \leq w$ . It follows that  $|\text{depend}_{k+1}(A)| \leq w(d+1)$ . Note that for  $k = n$ , we have  $\mathbf{y}_n = \mathbf{x}$  and therefore  $A_{(\mathbf{y}_n, \mathbf{a})} = \text{coeff}_A(\mathbf{x}^{\mathbf{a}})$  is a constant for every  $\mathbf{a}$ . Hence, the coefficient space has dimension one in this case, and thus  $|\text{span}_n(A)| = 1$ .

Now we are ready to construct an ROABP for  $A$ .

► **Lemma 2.5** ([21], Converse of Lemma 2.2). *Let  $A(\mathbf{x})$  be a polynomial of individual degree  $d$  with  $\mathbf{x} = (x_1, x_2, \dots, x_n)$ , such that for any  $1 \leq k \leq n$  and  $\mathbf{y}_k = (x_1, x_2, \dots, x_k)$ , we have*

$$\dim_{\mathbb{F}}\{A_{(\mathbf{y}_k, \mathbf{a})} \mid \mathbf{a} \in \{0, 1, \dots, d\}^k\} \leq w.$$

*Then there exists an ROABP of width  $w$  for  $A(\mathbf{x})$  in the variable order  $(x_1, x_2, \dots, x_n)$ .*

**Proof.** To keep the notation simple, we assume<sup>3</sup> that  $|\text{span}_k(A)| = w$  for each  $1 \leq k \leq n - 1$ . The argument would go through even when  $|\text{span}_k(A)| < w$ . Let  $\text{span}_k(A) = \{\mathbf{a}_{k,1}, \mathbf{a}_{k,2}, \dots, \mathbf{a}_{k,w}\}$  and  $\text{span}_n(A) = \{\mathbf{a}_{n,1}\}$ .

To prove the claim, we construct matrices  $D_1, D_2, \dots, D_n$ , where  $D_1 \in \mathbb{F}[x_1]^{1 \times w}$ ,  $D_n \in \mathbb{F}[x_n]^{w \times 1}$ , and  $D_i \in \mathbb{F}[x_i]^{w \times w}$ , for  $i = 2, \dots, n - 1$ , such that  $A(\mathbf{x}) = D_1 D_2 \cdots D_n$ . This representation shows that there is an ROABP of width  $w$  for  $A(\mathbf{x})$ .

<sup>3</sup> Assuming  $d + 1 \geq w$ ,  $\text{span}_k(A)$  can be made to have size  $= w$  for each  $k$ .

The matrices are constructed inductively such that for  $k = 1, 2, \dots, n-1$ ,

$$A(\mathbf{x}) = D_1 D_2 \cdots D_k [A(\mathbf{y}_k, \mathbf{a}_{k,1}) A(\mathbf{y}_k, \mathbf{a}_{k,2}) \cdots A(\mathbf{y}_k, \mathbf{a}_{k,w})]^T. \quad (4)$$

To construct  $D_1 \in \mathbb{F}[x_1]^{1 \times w}$ , consider the equation

$$A(\mathbf{x}) = \sum_{j=0}^d A(\mathbf{y}_1, j) x_1^j. \quad (5)$$

Recall that  $\text{depend}_1(A) = \{0, 1, \dots, d\}$ . By the definition of  $\text{span}_1(A)$ , every  $A(\mathbf{y}_1, j)$  is in the span of the  $A(\mathbf{y}_1, \mathbf{a})$ 's for  $\mathbf{a} \in \text{span}_1(A)$ . That is, there exists constants  $\{\gamma_{j,i}\}_{i,j}$  such that for all  $0 \leq j \leq d$  we have

$$A(\mathbf{y}_1, j) = \sum_{i=1}^w \gamma_{j,i} A(\mathbf{y}_1, \mathbf{a}_{1,i}). \quad (6)$$

From equations (5) and (6) we get,  $A(\mathbf{x}) = \sum_{i=1}^w \left( \sum_{j=0}^d \gamma_{j,i} x_1^j \right) A(\mathbf{y}_1, \mathbf{a}_{1,i})$ . Hence, we define  $D_1 = [D_{1,1} \ D_{1,2} \ \cdots \ D_{1,w}]$ , where  $D_{1,i} = \sum_{j=0}^d \gamma_{j,i} x_1^j$ , for all  $i \in [w]$ . Then we have

$$A = D_1 [A(\mathbf{y}_1, \mathbf{a}_{1,1}) A(\mathbf{y}_1, \mathbf{a}_{1,2}) \cdots A(\mathbf{y}_1, \mathbf{a}_{1,w})]^T. \quad (7)$$

To construct  $D_k \in \mathbb{F}[x_k]^{w \times w}$  for  $2 \leq k \leq n-1$ , we consider the equation

$$[A(\mathbf{y}_{k-1}, \mathbf{a}_{k-1,1}) \cdots A(\mathbf{y}_{k-1}, \mathbf{a}_{k-1,w})]^T = D_k [A(\mathbf{y}_k, \mathbf{a}_{k,1}) \cdots A(\mathbf{y}_k, \mathbf{a}_{k,w})]^T. \quad (8)$$

We know that for each  $1 \leq i \leq w$ ,

$$A(\mathbf{y}_{k-1}, \mathbf{a}_{k-1,i}) = \sum_{j=0}^d A(\mathbf{y}_k, (\mathbf{a}_{k-1,i}, j)) x_k^j. \quad (9)$$

Observe that  $(\mathbf{a}_{k-1,i}, j)$  is just an extension of  $\mathbf{a}_{k-1,i}$  and thus belongs to  $\text{depend}_k(A)$ . Hence, there exists a set of constants  $\{\gamma_{i,j,h}\}_{i,j,h}$  such that for all  $0 \leq j \leq d$  we have

$$A(\mathbf{y}_k, (\mathbf{a}_{k-1,i}, j)) = \sum_{h=1}^w \gamma_{i,j,h} A(\mathbf{y}_k, \mathbf{a}_{k,h}). \quad (10)$$

From equations (9) and (10), for each  $1 \leq i \leq w$  we get

$$A(\mathbf{y}_{k-1}, \mathbf{a}_{k-1,i}) = \sum_{h=1}^w \left( \sum_{j=0}^d \gamma_{i,j,h} x_k^j \right) A(\mathbf{y}_k, \mathbf{a}_{k,h}).$$

Hence, we can define  $D_k(i, h) = \sum_{j=0}^d \gamma_{i,j,h} x_k^j$ , for all  $i, h \in [w]$ . Then  $D_k$  is the desired matrix in equation (8).

Finally, we obtain  $D_n \in \mathbb{F}^{w \times 1}[x_n]$  in an analogous way. Instead of equation (8) we consider the equation

$$[A(\mathbf{y}_{n-1}, \mathbf{a}_{n-1,1}) \cdots A(\mathbf{y}_{n-1}, \mathbf{a}_{n-1,w})]^T = D'_n [A(\mathbf{y}_n, \mathbf{a}_{n,1})]. \quad (11)$$

Recall that  $A(\mathbf{y}_n, \mathbf{a}_{n,1}) \in \mathbb{F}$  is a constant that can be absorbed into the last matrix  $D'_n$ , i.e. we define  $D_n = D'_n A(\mathbf{y}_n, \mathbf{a}_{n,1})$ . Combining equations (7), (8), and (11), we get  $A(\mathbf{x}) = D_1 D_2 \cdots D_n$ .  $\blacktriangleleft$

Consider the polynomial  $P_k$  defined as the product of the first  $k$  matrices  $D_1, D_2, \dots, D_k$  from the above proof;  $P_k(\mathbf{y}_k) = D_1 D_2 \cdots D_k$ . We can write  $P_k$  as

$$P_k(\mathbf{y}_k) = \sum_{\mathbf{a} \in \{0,1,\dots,d\}^k} \text{coeff}_{P_k}(\mathbf{y}_k^{\mathbf{a}}) \mathbf{y}_k^{\mathbf{a}},$$

where  $\text{coeff}_{P_k}(\mathbf{y}_k^{\mathbf{a}})$  is a vector in  $\mathbb{F}^{1 \times w}$ . We will see next that it follows from the proof of Lemma 2.5 that the coefficient space of  $P_k$ , i.e.,  $\text{span}_{\mathbb{F}}\{\text{coeff}_{P_k}(\mathbf{y}_k^{\mathbf{a}}) \mid \mathbf{a} \in \{0,1,\dots,d\}^k\}$  has full rank  $w$ .

► **Corollary 2.6** (Full Rank Coefficient Space). *Let  $D_1, D_2, \dots, D_n$  be the matrices constructed in the proof of Lemma 2.5 with  $A = D_1 D_2 \cdots D_n$ . Let  $\text{span}_k(A) = \{\mathbf{a}_{k,1}, \mathbf{a}_{k,2}, \dots, \mathbf{a}_{k,w}\}$ . For  $k \in [n]$ , define the polynomial  $P_k(\mathbf{y}_k) = D_1 D_2 \cdots D_k$ .*

*Then for any  $\ell \in [w]$ , we have  $\text{coeff}_{P_k}(\mathbf{y}_k^{\mathbf{a}_{k,\ell}}) = \mathbf{e}_{\ell}$ , where  $\mathbf{e}_{\ell}$  is the  $\ell$ -th elementary unit vector,  $\mathbf{e}_{\ell} = (0, \dots, 0, 1, 0, \dots, 0)$  of length  $w$ , with a one at position  $\ell$ , and zero at all other positions. Hence, the coefficient space of  $P_k$  has full rank  $w$ .*

**Proof.** In the construction of the matrices  $D_k$  in the proof of Lemma 2.5, consider the special case in equations (6) and (10) that the exponent  $(\mathbf{a}_{k-1,i}, j)$  is in  $\text{span}_k(A)$ , say  $(\mathbf{a}_{k-1,i}, j) = \mathbf{a}_{k,\ell} \in \text{span}_k(A)$ . Then the  $\gamma$ -vector to express  $A_{(\mathbf{y}_k, (\mathbf{a}_{k-1,i}, j))}$  in equation (6) and (10) can be chosen to be  $\mathbf{e}_{\ell}$ , i.e.  $(\gamma_{i,j,h})_h = \mathbf{e}_{\ell}$ . By the definition of matrix  $D_k$ , vector  $\mathbf{e}_{\ell}$  becomes the  $i$ -th row of  $D_k$  for the exponent  $j$ , i.e.,  $\text{coeff}_{D_k(i,\cdot)}(x_k^j) = \mathbf{e}_{\ell}$ .

This shows the claim for  $k = 1$ . For larger  $k$ , it follows by induction because for  $(\mathbf{a}_{k-1,i}, j) = \mathbf{a}_{k,\ell}$  we have  $\text{coeff}_{P_k}(\mathbf{y}_k^{\mathbf{a}_{k,\ell}}) = \text{coeff}_{P_{k-1}}(\mathbf{y}_{k-1}^{\mathbf{a}_{k-1,i}}) \text{coeff}_{D_k}(x_k^j)$ . ◀

### 3 Whitebox Identity Testing

We will use the characterization of ROABPs provided by Lemmas 2.2 and 2.5 in Section 3.1 to design a polynomial-time algorithm to check if two given ROABPs are equivalent. This is the same problem as to check whether the sum of two ROABPs is zero. In Section 3.2, we extend the test to check whether the sum of constantly many ROABPs is zero.

#### 3.1 Equivalence of two ROABPs

Let  $A(\mathbf{x})$  and  $B(\mathbf{x})$  be two polynomials of individual degree  $d$ , given by two ROABPs. If the two ROABPs have the same variable order then one can combine them into a single ROABP which computes their difference. Then one can apply the test for one ROABP (whitebox [22], blackbox [3]). So, the problem is non-trivial only when the two ROABPs have different variable order. W.l.o.g. we assume that  $A$  has order  $(x_1, x_2, \dots, x_n)$ . Let  $w$  bound the width of both ROABPs. In this section we prove that we can find out in polynomial time whether  $A(\mathbf{x}) = B(\mathbf{x})$ .

► **Theorem 3.1.** *The equivalence of two ROABPs can be tested in polynomial time.*

The idea is to determine the characterizing set of dependencies among the partial derivative polynomials of  $A$ , and verify that the same dependencies hold for the corresponding partial derivative polynomials of  $B$ . By Lemma 2.5, these dependencies essentially define an ROABP. Hence, our algorithm is to construct an ROABP for  $B$  in the variable order of  $A$ . Then it suffices to check whether we get the same ROABP, that is, all the matrices  $D_1, D_2, \dots, D_n$  constructed in the proof of Lemma 2.5 are the same for  $A$  and  $B$ . We give some more details.

### Construction of $\text{span}_k(A)$

Let  $A(\mathbf{x}) = D_1(x_1)D_2(x_2) \cdots D_n(x_n)$  of width  $w$ . We give an iterative construction, starting from  $\text{span}_0(A) = \{\epsilon\}$ . Let  $1 \leq k \leq n$ . By definition,  $\text{depend}_k(A)$  consists of all possible one-step extensions of  $\text{span}_{k-1}(A)$ . Let  $\mathbf{b} = (b_1, b_2, \dots, b_k) \in \{0, 1, \dots, d\}^k$ . Define

$$C_{\mathbf{b}} = \prod_{i=1}^k \text{coeff}_{D_i}(x_i^{b_i}).$$

Recall that  $\text{coeff}_{D_1}(x_1^{b_1}) \in \mathbb{F}^{1 \times w}$  and  $\text{coeff}_{D_i}(x_i^{b_i}) \in \mathbb{F}^{w \times w}$ , for  $2 \leq i \leq k$ . Therefore  $C_{\mathbf{b}} \in \mathbb{F}^{1 \times w}$  for  $k < n$ . Since  $D_n \in \mathbb{F}^{w \times 1}$ , we have  $C_{\mathbf{b}} \in \mathbb{F}$  for  $k = n$ . By equation (3), we have

$$A_{(\mathbf{y}_k, \mathbf{b})} = C_{\mathbf{b}} D_{k+1} \cdots D_n. \quad (12)$$

Consider the set of vectors  $\mathcal{D}_k = \{C_{\mathbf{b}} \mid \mathbf{b} \in \text{depend}_k(A)\}$ . This set has dimension bounded by  $w$  since the width of  $A$  is  $w$ . Hence, we can determine a set  $\mathcal{S}_k \subseteq \mathcal{D}_k$  of size  $\leq w$  such that  $\mathcal{S}_k$  spans  $\mathcal{D}_k$ . Thus we can take  $\text{span}_k(A) = \{\mathbf{a} \mid C_{\mathbf{a}} \in \mathcal{S}_k\}$ . Then, for any  $\mathbf{b} \in \text{depend}_k(A)$ , vector  $C_{\mathbf{b}}$  is a linear combination

$$C_{\mathbf{b}} = \sum_{\mathbf{a} \in \text{span}_k(A)} \gamma_{\mathbf{a}} C_{\mathbf{a}}.$$

Recall that  $|\text{depend}_k(A)| \leq w(d+1)$ , i.e. this is a small set. Therefore we can efficiently compute the coefficients  $\gamma_{\mathbf{a}}$  for every  $\mathbf{b} \in \text{depend}_k(A)$ . Note that by equation (12) we have the same dependencies for the polynomials  $A_{(\mathbf{y}_k, \mathbf{b})}$ . That is, with the same coefficients  $\gamma_{\mathbf{a}}$ , we can write

$$A_{(\mathbf{y}_k, \mathbf{b})} = \sum_{\mathbf{a} \in \text{span}_k(A)} \gamma_{\mathbf{a}} A_{(\mathbf{y}_k, \mathbf{a})}. \quad (13)$$

### Verifying the dependencies for $B$

We want to verify that the dependencies in equation (13) computed for  $A$  hold as well for  $B$ , i.e. that for all  $k \in [n]$  and  $\mathbf{b} \in \text{depend}_k(A)$ ,

$$B_{(\mathbf{y}_k, \mathbf{b})} = \sum_{\mathbf{a} \in \text{span}_k(A)} \gamma_{\mathbf{a}} B_{(\mathbf{y}_k, \mathbf{a})}. \quad (14)$$

Recall that  $\mathbf{y}_k = (x_1, x_2, \dots, x_k)$  and the ROABP for  $B$  has a different variable order. By Lemma 2.3, every polynomial  $B_{(\mathbf{y}_k, \mathbf{a})}$  has an ROABP of width  $w$  and the same order on the remaining variables as the one given for  $B$ . It follows that each of the  $w+1$  polynomials that occur in equation (14) has an ROABP of width  $w$  and the same variable order. Hence, we can construct *one* ROABP for the polynomial

$$B_{(\mathbf{y}_k, \mathbf{b})} - \sum_{\mathbf{a} \in \text{span}_k(A)} \gamma_{\mathbf{a}} B_{(\mathbf{y}_k, \mathbf{a})}. \quad (15)$$

Simply identify all the start nodes and all the end nodes and put the appropriate constants  $\gamma_{\mathbf{a}}$  to the weights. Then we get an ROABP of width  $w(w+1)$ . In order to verify equation (14), it suffices to make a zero-test for this ROABP. This can be done in polynomial time [22].

### Correctness

Clearly, if equation (14) fails to hold for some  $k$  and  $\mathbf{b}$ , then  $A \neq B$ . So assume that equation (14) holds for all  $k$  and  $\mathbf{b}$ .

Recall Lemma 2.5 and its proof. There we constructed an ROABP just from the characterizing dependencies of the given polynomial. Hence, the construction applied to  $B$  will give an ROABP of width  $w$  for  $B$  with the same variable order  $(x_1, x_2, \dots, x_n)$  as for  $A$ . The matrices  $D_k$  will be the same as for  $A$  because their definition uses only the dependencies provided by equation (14), and they are the same as for  $A$  in equation (13).

Note that when we construct the last matrix  $D_n$  by equation (11), for  $A$  we have  $A(\mathbf{x}) = D_1 D_2 \cdots D_n$ , where  $D_n = D'_n A_{(\mathbf{y}_n, \mathbf{a}_{n,1})}$ . The dependencies define matrix  $D'_n$ . Therefore, for  $B$  we will obtain  $B(\mathbf{x}) = D_1 D_2 \cdots D'_n B_{(\mathbf{y}_n, \mathbf{a}_{n,1})}$ . Since we also check that we get the same matrix  $D_n$  for  $A$  and  $B$ , we also have  $A_{(\mathbf{y}_n, \mathbf{a}_{n,1})} = B_{(\mathbf{y}_n, \mathbf{a}_{n,1})}$ , and therefore  $A(\mathbf{x}) = B(\mathbf{x})$ . This proves Theorem 3.1.

### 3.2 Sum of constantly many ROABPs

Let  $A_1(\mathbf{x}), A_2(\mathbf{x}), \dots, A_c(\mathbf{x})$  be polynomials of individual degree  $d$ , given by  $c$  ROABPs. Our goal is to test whether  $A_1 + A_2 + \dots + A_c = 0$ . Here again, the question is interesting only when the ROABPs have different variable orders. We show how to reduce the problem to the case of the equivalence of two ROABPs from the previous section. For constant  $c$  this will lead to a polynomial-time test.

We start by rephrasing the problem as an equivalence test. Let  $A = -A_1$  and  $B = A_2 + A_3 + \dots + A_c$ . Then the problem has become to check whether  $A = B$ . Since  $A$  is computed by a single ROABP, we can use the same approach as in Section 3.1. Hence, we get again the dependencies from equation (13) for  $A$ . Next, we have to verify these dependencies for  $B$ , i.e. equation (14). Now,  $B$  is not given by a single ROABP, but is a sum of  $c - 1$  ROABPs. For every  $k \in [n]$  and  $\mathbf{b} \in \text{depend}_k(A)$ , define the polynomial  $Q = B_{(\mathbf{y}_k, \mathbf{b})} - \sum_{\mathbf{a} \in \text{span}_k(A)} \gamma_{\mathbf{a}} B_{(\mathbf{y}_k, \mathbf{a})}$ . By the definition of  $B$  we have

$$Q = \sum_{i=2}^c \left( A_{i(\mathbf{y}_k, \mathbf{b})} - \sum_{\mathbf{a} \in \text{span}_k(A)} \gamma_{\mathbf{a}} A_{i(\mathbf{y}_k, \mathbf{a})} \right). \tag{16}$$

As explained in the previous section for equation (15), for each summand in equation (16) we can construct an ROABP of width  $w(w + 1)$ . Thus,  $Q$  can be written as a sum of  $c - 1$  ROABPs, each having width  $w(w + 1)$ . To test whether  $Q = 0$ , we recursively use the same algorithm for the sum of  $c - 1$  ROABPs. The recursion ends when  $c = 2$ . Then we directly use the algorithm from Section 3.1.

To bound the running time of the algorithm, let us see how many dependencies we need to verify. There is one dependency for every  $k \in [n]$  and every  $\mathbf{b} \in \text{depend}_k(A)$ . Since  $|\text{depend}_k(A)| \leq w(d + 1)$ , the total number of dependencies verified is  $\leq nw(d + 1)$ . Thus, we get the following recursive formula for  $T(c, w)$ , the time complexity for testing zeroness of the sum of  $c \geq 2$  ROABPs, each having width  $w$ . For  $c = 2$ , we have  $T(2, w) = \text{poly}(n, d, w)$ , and for  $c > 2$ ,

$$T(c, w) = nw(d + 1) \cdot T(c - 1, w(w + 1)) + \text{poly}(n, d, w).$$

As solution, we get  $T(c, w) = w^{O(2^c)} \text{poly}(n^c, d^c)$ , i.e. polynomial time for constant  $c$ .

► **Theorem 3.2.** *Let  $A(\mathbf{x})$  be an  $n$ -variate polynomial of individual degree  $d$ , computed by a sum of  $c$  ROABPs of width  $w$ . Then there is a PIT for  $A(\mathbf{x})$  that works in time  $w^{O(2^c)}(nd)^{O(c)}$ .*

## 4 Blackbox Identity Testing

In this section, we extend the blackbox PIT of Agrawal et. al [3] for one ROABP to the sum of constantly many ROABPs. In the blackbox model we are only allowed to evaluate a polynomial at various points. Hence, for PIT, our task is to construct a hitting-set.

► **Definition 4.1.** A set  $H = H(n, d, w) \subseteq \mathbb{F}^n$  is a *hitting-set for ROABPs*, if for every nonzero  $n$ -variate polynomial  $A(\mathbf{x})$  of individual degree  $d$  that can be computed by ROABPs of width  $w$ , there is a point  $\mathbf{a} \in H$  such that  $A(\mathbf{a}) \neq 0$ .

For polynomials computed by a sum of  $c$  ROABPs, a hitting-set is defined similarly. Here,  $H = H(n, d, w, c)$  additionally depends on  $c$ .

For a hitting-set to exist, we will need enough points in the underlying field  $\mathbb{F}$ . Henceforth, we will assume that the field  $\mathbb{F}$  is large enough such that the constructions below go through (see [1] for constructing large  $\mathbb{F}$ ). To construct a hitting-set for a sum of ROABPs we use the concept of *low support rank concentration* defined by Agrawal, Saha, and Saxena [4]. A polynomial  $A(\mathbf{x})$  has low support concentration if the coefficients of its monomials of low support span the coefficients of all the monomials.

► **Definition 4.2** ([4]). A polynomial  $A(\mathbf{x})$  has  *$\ell$ -support concentration* if for all monomials  $\mathbf{x}^{\mathbf{a}}$  of  $A(\mathbf{x})$  we have

$$\text{coeff}_A(\mathbf{x}^{\mathbf{a}}) \in \text{span}_{\mathbb{F}}\{\text{coeff}_A(\mathbf{x}^{\mathbf{b}}) \mid \text{supp}(\mathbf{b}) < \ell\}.$$

The above definition applies to polynomials over any  $\mathbb{F}$ -vector space, e.g.  $\mathbb{F}[\mathbf{x}]$ ,  $\mathbb{F}^w[\mathbf{x}]$  or  $\mathbb{F}^{w \times w}[\mathbf{x}]$ . If  $A(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$  is a non-zero polynomial that has  $\ell$ -support concentration, then there are nonzero coefficients of support  $< \ell$ . Then the assignments of support  $< \ell$  are a hitting-set for  $A(\mathbf{x})$ .

► **Lemma 4.3** ([4]). *For  $n, d, \ell$ , the set  $H = \{\mathbf{h} \in \{0, \beta_1, \dots, \beta_d\}^n \mid \text{supp}(\mathbf{h}) < \ell\}$  of size  $(nd)^{O(\ell)}$  is a hitting-set for all  $n$ -variate  $\ell$ -concentrated polynomials  $A(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$  of individual degree  $d$ , where  $\{\beta_i\}_i$  are distinct nonzero elements in  $\mathbb{F}$ .*

Hence, when we have low support concentration, this solves blackbox PIT. However, not every polynomial has a low support concentration, for example  $A(\mathbf{x}) = x_1 x_2 \cdots x_n$  is not  $n$ -concentrated. However, Agrawal, Saha, and Saxena [4] showed that low support concentration can be achieved through an appropriate shift of the variables.

► **Definition 4.4.** Let  $A(\mathbf{x})$  be an  $n$ -variate polynomial and  $\mathbf{f} = (f_1, f_2, \dots, f_n) \in \mathbb{F}^n$ . The polynomial  $A$  shifted by  $\mathbf{f}$  is  $A(\mathbf{x} + \mathbf{f}) = A(x_1 + f_1, x_2 + f_2, \dots, x_n + f_n)$ .

Note that a shift is an invertible process. Therefore it preserves the coefficient space of a polynomial.

In the above example, we shift every variable by 1. That is, we consider  $A(\mathbf{x} + \mathbf{1}) = (x_1 + 1)(x_2 + 1) \cdots (x_n + 1)$ . Observe that  $A(\mathbf{x} + \mathbf{1})$  has 1-support concentration. Agrawal, Saha, and Saxena [4] provide an efficient shift that achieves low support concentration for polynomials computed by *set-multilinear depth-3 circuits*. Forbes, Saptharishi and Shpilka [9] extended their result to polynomials computed by ROABPs. However their cost is exponential



in the individual degree of the polynomial. Any efficient shift for ROABPs will suffice for our purposes. Here, we will give a new shift for ROABPs with quasi-polynomial cost. Namely, in Theorem 5.6 below we present a shift polynomial  $\mathbf{f}(t) \in \mathbb{F}[t]^n$  in one variable  $t$  of degree  $(ndw)^{O(\log n)}$  that can be computed in time  $(ndw)^{O(\log n)}$ . It has the property that for every  $n$ -variate polynomial  $A(\mathbf{x}) \in \mathbb{F}^{w \times w}[\mathbf{x}]$  of individual degree  $d$  that can be computed by an ROABP of width  $w$ , the shifted polynomial  $A(\mathbf{x} + \mathbf{f}(t))$  has  $O(\log w)$ -concentration. We can plug in as many values for  $t \in \mathbb{F}$  as the degree of  $\mathbf{f}(t)$ , i.e.  $(ndw)^{O(\log n)}$  many. For at least one value of  $t$ , the shift  $\mathbf{f}(t)$  will  $O(\log w)$ -concentrate  $A(\mathbf{x} + \mathbf{f}(t))$ . That is, we consider  $\mathbf{f}(t)$  as a family of shifts. The same shift also works when the ROABP computes a polynomial in  $\mathbb{F}[\mathbf{x}]$  or  $\mathbb{F}^{1 \times w}[\mathbf{x}]$ .

The rest of the paper is organized as follows. The construction of a shift to obtain low support concentration for single ROABPs is postponed to Section 5. We start in Section 4.1 to show how the shift for a single ROABP can be applied to obtain a shift for the sum of constantly many ROABPs.

### 4.1 Sum of ROABPs

Let polynomial  $A \in \mathbb{F}[\mathbf{x}]$  of individual degree  $d$  have an ROABP of width  $w$ , with variable order  $(x_1, x_2, \dots, x_n)$ . Let  $B \in \mathbb{F}[\mathbf{x}]$  be another polynomial. We start by reconsidering the whitebox test from the previous section. The dependency equations (13) and (14) were used to construct an ROABP for  $B \in \mathbb{F}[\mathbf{x}]$  in the same variable order as for  $A$ , and the same width. If this succeeds, then the polynomial  $A + B$  has one ROABP of width  $2w$ . Since there is already a blackbox PIT for one ROABP [3], we are done in this case. Hence, the interesting case that remains is when  $B$  does not have an ROABP of width  $w$  in the variable order of  $A$ .

Let  $k \in [n]$  be the first index such that the dependency equations (13) for  $A$  do not carry over to  $B$  as in equation (14). In the following Lemma 4.5 we decompose  $A$  and  $B$  into a common part up to layer  $k$ , and the remaining different parts. That is, for  $\mathbf{y}_k = (x_1, x_2, \dots, x_k)$  and  $\mathbf{z}_k = (x_{k+1}, \dots, x_n)$ , we obtain  $A = RP$  and  $B = RQ$ , where  $R \in \mathbb{F}[\mathbf{y}_k]^{1 \times w'}$  and  $P, Q \in \mathbb{F}[\mathbf{z}_k]^{w' \times 1}$ , for some  $w' \leq w(d + 1)$ . The construction also implies that the coefficient space of  $R$  has full rank  $w'$ . Since the dependency equations (13) for  $A$  do not fulfill equation (14) for  $B$ , we get a constant vector  $\Gamma \in \mathbb{F}^{1 \times w'}$  such that  $\Gamma P = 0$  but  $\Gamma Q \neq 0$ . From these properties we will see in Lemma 4.6 below that we get low support concentration for  $A + B$  when we use the shift constructed in Section 5 for one ROABP.

► **Lemma 4.5 (Common ROABP  $R$ ).** *Let  $A(\mathbf{x})$  be polynomial of individual degree  $d$ , computed by a ROABP of width  $w$  in variable order  $(x_1, x_2, \dots, x_n)$ . Let  $B(\mathbf{x})$  be another polynomial for which there does not exist an ROABP of width  $w$  in the same variable order.*

*Then there exists a  $k \in [n]$  such that for some  $w' \leq w(d + 1)$ , there are polynomials  $R \in \mathbb{F}[\mathbf{y}_k]^{1 \times w'}$  and  $P, Q \in \mathbb{F}[\mathbf{z}_k]^{w' \times 1}$ , such that*

1.  $A = RP$  and  $B = RQ$ ,
2. *there exists a vector  $\Gamma \in \mathbb{F}^{1 \times w'}$  with  $\text{supp}(\Gamma) \leq w + 1$  such that  $\Gamma P = 0$  and  $\Gamma Q \neq 0$ ,*
3. *the coefficient space of  $R$  has full rank  $w'$ .*

**Proof.** Let  $D_1, D_2, \dots, D_n$  be the matrices constructed in Lemma 2.5 for  $A$ . Assume again w.l.o.g. that  $\text{span}_k(A) = \{\mathbf{a}_{k,1}, \mathbf{a}_{k,2}, \dots, \mathbf{a}_{k,w}\}$  has size  $w$  for each  $1 \leq k \leq n - 1$ , and  $\text{span}_n(A) = \{\mathbf{a}_{n,1}\}$ . Then we have  $D_1 \in \mathbb{F}^{1 \times w}[x_1]$ ,  $D_n \in \mathbb{F}^{w \times 1}[x_n]$  and  $D_i \in \mathbb{F}^{w \times w}[x_i]$ , for  $2 \leq i \leq n - 1$ .

In the proof of Lemma 2.5 we consider the dependency equations for  $A$  and carry them over to  $B$ . By the assumption of the lemma, there is no ROABP of width  $w$  for  $B$  now.

Therefore there is a smallest  $k \in [n]$  where a dependency for  $A$  is not followed by  $B$ . That is, the coefficients  $\gamma_{\mathbf{a}}$  computed for equation (13) do not fulfill equation (14) for  $B$ . Since the dependencies carry over up to this point, the construction of the matrices  $D_1, D_2, \dots, D_{k-1}$  work out fine for  $B$ . Hence, by equation (4), we can write

$$A(\mathbf{x}) = D_1 D_2 \cdots D_{k-1} [A(\mathbf{y}_{k-1}, \mathbf{a}_{k-1,1}) A(\mathbf{y}_{k-1}, \mathbf{a}_{k-1,2}) \cdots A(\mathbf{y}_{k-1}, \mathbf{a}_{k-1,w})]^T \quad (17)$$

$$B(\mathbf{x}) = D_1 D_2 \cdots D_{k-1} [B(\mathbf{y}_{k-1}, \mathbf{a}_{k-1,1}) B(\mathbf{y}_{k-1}, \mathbf{a}_{k-1,2}) \cdots B(\mathbf{y}_{k-1}, \mathbf{a}_{k-1,w})]^T \quad (18)$$

Since the difference between  $A$  and  $B$  occurs at  $x_k$ , we consider all possible extensions from  $x_{k-1}$ . That is, by equation (9), for every  $i \in [w]$  we have

$$A_{(\mathbf{y}_{k-1}, \mathbf{a}_{k-1,i})} = \sum_{j=0}^d A_{(\mathbf{y}_k, (\mathbf{a}_{k-1,i,j}))} x_k^j. \quad (19)$$

Recall that our goal is to decompose polynomial  $A$  into  $A = RP$ . We first define polynomial  $P$  as the vector of coefficient polynomials of all the one-step extensions of  $\text{span}_{k-1}(A)$ , i.e.,  $P = (A_{(\mathbf{y}_k, (\mathbf{a}_{k-1,i,j}))})_{1 \leq i \leq w, 0 \leq j \leq d}$  is of length  $w' = w(d+1)$ . Written explicitly, this is

$$P = [A_{(\mathbf{y}_k, (\mathbf{a}_{k-1,1,0}))} \cdots A_{(\mathbf{y}_k, (\mathbf{a}_{k-1,1,d}))} \cdots A_{(\mathbf{y}_k, (\mathbf{a}_{k-1,w,0}))} \cdots A_{(\mathbf{y}_k, (\mathbf{a}_{k-1,w,d}))}]^T.$$

To define  $R \in \mathbb{F}[\mathbf{y}_k]^{1 \times w'}$ , let  $I_w$  be the  $w \times w$  identity matrix. Define matrix  $E_k \in \mathbb{F}[x_k]^{w \times w'}$  as the tensor product

$$E_k = I_w \otimes [x_k^0 \ x_k^1 \ \cdots \ x_k^d].$$

From equation (19) we get that

$$[A_{(\mathbf{y}_{k-1}, \mathbf{a}_{k-1,1})} \cdots A_{(\mathbf{y}_{k-1}, \mathbf{a}_{k-1,w})}]^T = E_k P.$$

Thus, equation (17) can be written as  $A(\mathbf{x}) = D_1 D_2 \cdots D_{k-1} E_k P$ . Hence, when we define

$$R(\mathbf{y}_k) = D_1 D_2 \cdots D_{k-1} E_k$$

then we have  $A = RP$  as desired. By an analogous argument we get  $B = RQ$  for  $Q = (B_{(\mathbf{y}_k, (\mathbf{a}_{k-1,i,j}))})_{1 \leq i \leq w, 0 \leq j \leq d}$ .

For the second claim of the lemma let  $\mathbf{b} \in \text{depend}_k(A)$  such that the dependency equation (13) for  $A$  is fulfilled, but not equation (14) for  $B$ . Define  $\Gamma \in \mathbb{F}^{1 \times w'}$  to be the vector that has the values  $\gamma_{\mathbf{a}}$  used in equation (13) at the position where  $P$  has entry  $A_{(\mathbf{y}_k, \mathbf{a})}$ , and zero at all other positions. Then  $\text{supp}(\Gamma) \leq w+1$  and we have  $\Gamma P = 0$  and  $\Gamma Q \neq 0$ .

It remains to show that the coefficient space of  $R$  has full rank. By Corollary 2.6, the coefficient space of  $D_1 D_2 \cdots D_{k-1}$  has full rank  $w$ . Namely, for any  $\ell \in [w]$ , the coefficient of the monomial  $\mathbf{y}_{k-1}^{\mathbf{a}_{k-1,\ell}}$  is  $\mathbf{e}_\ell$ , the  $\ell$ -th standard unit vector. Therefore the coefficient of  $R(\mathbf{y}_k) = D_1 D_2 \cdots D_{k-1} E_k$  at monomial  $\mathbf{y}_k^{(\mathbf{a}_{k-1,\ell,j})}$  is

$$\text{coeff}_R(\mathbf{y}_k^{\mathbf{a}_{k-1,\ell,j}}) = \mathbf{e}_\ell \text{coeff}_{E_k}(x_k^j),$$

for  $1 \leq \ell \leq w$  and  $0 \leq j \leq d$ . By the definition of  $E_k$ , we get  $\text{coeff}_R(\mathbf{y}_k^{\mathbf{a}_{k-1,\ell,j}}) = e_{(\ell-1)(d+1)+j+1}$ . Thus, the coefficient space of  $R$  has full rank  $w'$ . ◀

Lemma 4.5 provides the technical tool to obtain low support concentration for the sum of several ROABPs by the shift developed for a single ROABP. We start with the case of the sum of two ROABPs.

► **Lemma 4.6.** *Let  $A(\mathbf{x})$  and  $B(\mathbf{x})$  be two  $n$ -variate polynomials of individual degree  $d$ , each computed by an ROABP of width  $w$ . Define  $W_{w,2} = (d+1)(2w)^2$  and  $\ell_{w,2} = \log(W_{w,2}^2 + 1)$ . Let  $\mathbf{f}_{w,2}(t) \in \mathbb{F}[t]^n$  be a shift that  $\ell_{w,2}$ -concentrates any polynomial (or matrix polynomial) that is computed by an ROABP of width  $\leq W_{w,2}$ .*

*Then  $(A+B)' = (A+B)(\mathbf{x} + \mathbf{f}_{w,2})$  is  $2\ell_{w,2}$ -concentrated.*

**Proof.** If  $B$  can be computed by an ROABP of width  $w$  in the same variable order as the one for  $A$ , then there is an ROABP of width  $2w$  that computes  $A+B$ . In this case the lemma follows because  $2w \leq W_{w,2}$ . So let us assume that there is no such ROABP for  $B$ . Thus the assumption from Lemma 4.5 is fulfilled. Hence, we have a decomposition of  $A$  and  $B$  at the  $k$ -th layer into  $A(\mathbf{x}) = R(\mathbf{y}_k)P(\mathbf{z}_k)$  and  $B(\mathbf{x}) = R(\mathbf{y}_k)Q(\mathbf{z}_k)$ , and there is a vector  $\Gamma \in \mathbb{F}^{1 \times w'}$  such that  $\Gamma P = 0$  and  $\Gamma Q \neq 0$ , where  $w' = (d+1)w$  and  $\text{supp}(\Gamma) \leq w+1$ .

Define  $R', P', Q'$  as the polynomials  $R, P, Q$  shifted by  $\mathbf{f}_{w,2}$ , respectively. Since  $\Gamma P = 0$ , we also have  $\Gamma P' = 0$ .

By the definition of  $R$ , there is an ROABP of width  $w'$  that computes  $R$ . Since  $w' \leq W_{w,2}$ , polynomial  $R'$  is  $\ell_{w,2}$ -concentrated by the assumption of the lemma.

We argue that also  $\Gamma Q'$  is  $\ell_{w,2}$ -concentrated: let  $Q = [Q_1 Q_2 \cdots Q_{w'}]^T \in \mathbb{F}[\mathbf{z}_k]^{w' \times 1}$ . By Lemma 2.3, from the ROABP for  $B$  we get an ROABP for each  $Q_i$  of the same width  $w$  and the same variable order. Therefore we can combine them into one ROABP that computes  $\Gamma Q = \sum_{i=1}^{w'} \gamma_i Q_i$ . Its width is  $w(w+1)$  because  $\text{supp}(\Gamma) \leq w+1$ . Since  $w(w+1) \leq W_{w,2}$ , polynomial  $\Gamma Q$  is  $\ell_{w,2}$ -concentrated.

Since  $\Gamma Q \neq 0$  and  $\Gamma Q'$  is  $\ell_{w,2}$ -concentrated, there exists at least one  $\mathbf{b} \in \{0, 1, \dots, d\}^{n-k}$  with  $\text{supp}(\mathbf{b}) < \ell_{w,2}$  such that  $\Gamma \text{coeff}_{Q'}(\mathbf{z}_k^{\mathbf{b}}) \neq 0$ . Because  $\Gamma P = 0$ , we have  $\Gamma \text{coeff}_{P'}(\mathbf{z}_k^{\mathbf{b}}) = 0$ , and therefore

$$\Gamma \text{coeff}_{P'+Q'}(\mathbf{z}_k^{\mathbf{b}}) \neq 0. \tag{20}$$

Recall that the coefficient space of  $R$  has full rank  $w'$ . Since a shift preserves the coefficient space, also  $R'$  has a full rank coefficient space. Because  $R'$  is  $\ell_{w,2}$ -concentrated, already the coefficients of the  $< \ell_{w,2}$ -support monomials of  $R'$  have full rank  $w'$ . That is, for  $M_{\ell_{w,2}} = \{\mathbf{a} \in \{0, 1, \dots, d\}^k \mid \text{supp}(\mathbf{a}) < \ell_{w,2}\}$ , we have  $\text{rank}_{\mathbb{F}(t)}\{\text{coeff}_{R'}(\mathbf{y}_k^{\mathbf{a}}) \mid \mathbf{a} \in M_{\ell_{w,2}}\} = w'$ . Therefore, we can express  $\Gamma$  as a linear combination of these coefficients,

$$\Gamma = \sum_{\mathbf{a} \in M_{\ell_{w,2}}} \alpha_{\mathbf{a}} \text{coeff}_{R'}(\mathbf{y}_k^{\mathbf{a}}),$$

where  $\alpha_{\mathbf{a}}$  is a rational function in  $\mathbb{F}(t)$ , for  $\mathbf{a} \in M_{\ell_{w,2}}$ . Hence, from equation (20) we get

$$\begin{aligned} \Gamma \text{coeff}_{(P'+Q')}(\mathbf{z}_k^{\mathbf{b}}) &= \left( \sum_{\mathbf{a} \in M_{\ell_{w,2}}} \alpha_{\mathbf{a}} \text{coeff}_{R'}(\mathbf{y}_k^{\mathbf{a}}) \right) \text{coeff}_{P'+Q'}(\mathbf{z}_k^{\mathbf{b}}) \\ &= \sum_{\mathbf{a} \in M_{\ell_{w,2}}} \alpha_{\mathbf{a}} \text{coeff}_{R'(P'+Q')}(\mathbf{y}_k^{\mathbf{a}} \mathbf{z}_k^{\mathbf{b}}) \\ &= \sum_{\mathbf{a} \in M_{\ell_{w,2}}} \alpha_{\mathbf{a}} \text{coeff}_{(A+B)' }(\mathbf{x}^{(\mathbf{a}, \mathbf{b})}) \\ &\neq 0. \end{aligned}$$

Since  $\text{supp}(\mathbf{a}, \mathbf{b}) = \text{supp}(\mathbf{a}) + \text{supp}(\mathbf{b}) < 2\ell_{w,2}$ , it follows that there is a monomial in  $(A+B)'$  of support  $< 2\ell_{w,2}$  with a nonzero coefficient. In other words,  $(A+B)'$  is  $2\ell_{w,2}$ -concentrated. ◀

In Section 5, Theorem 5.6, we will show that the shift polynomial  $\mathbf{f}_{w,2}(t) \in \mathbb{F}[t]^n$  used in Lemma 4.6 can be computed in time  $(ndw)^{O(\log n)}$ . The degree of  $\mathbf{f}_{w,2}(t)$  has the same bound. Recall that when we say that we shift by  $\mathbf{f}_{w,2}(t)$ , we actually mean that we plug in values for  $t$  up to the degree of  $\mathbf{f}_{w,2}(t)$ . That is, we have a family of  $(ndw)^{O(\log n)}$  shifts, and at least one of them will give low support concentration. By Lemma 4.3, we get for each  $t$ , a potential hitting-set  $H_t$  of size  $(nd)^{O(\ell_{w,2})} = (nd)^{O(\log dw)}$ ,

$$H_t = \{\mathbf{h} + \mathbf{f}(t) \mid \mathbf{h} \in \{0, \beta_1, \dots, \beta_d\}^n \text{ and } \text{supp}(\mathbf{h}) < 2\ell_{w,2}\}.$$

The final hitting-set is the union of all these sets, i.e.  $H = \bigcup_t H_t$ , where  $t$  takes  $(ndw)^{O(\log n)}$  distinct values. Hence, we have the following main result.

► **Theorem 4.7.** *Given  $n, d, w$ , in time  $(ndw)^{O(\log ndw)}$  one can construct a hitting-set for all  $n$ -variate polynomials of individual degree  $d$ , that can be computed by a sum of two ROABPs of width  $w$ .*

We extend Lemma 4.6 to the sum of  $c$  ROABPs.

► **Lemma 4.8.** *Let  $A = A_1 + A_2 + \dots + A_c$ , where the  $A_i$ 's are  $n$ -variate polynomials of individual degree  $d$ , each computed by an ROABP of width  $w$ . Define  $W_{w,c} = (d+1)(2w)^{2^{c-1}}$  and  $\ell_{w,c} = \log(W_{w,c}^2 + 1)$ . Let  $\mathbf{f}_{w,c}(t) \in \mathbb{F}[t]^n$  be a shift that  $\ell_{w,c}$ -concentrates any polynomial (or matrix polynomial) that is computed by an ROABP of width  $W_{w,c}$ .*

*Then  $A' = A(\mathbf{x} + \mathbf{f}_{w,c})$  is  $c\ell_{w,c}$ -concentrated.*

**Proof.** The proof is by induction on  $c$ . Lemma 4.6 provides the base case  $c = 2$ . For the induction step let  $c \geq 3$ . We follow the proof of Lemma 4.6 with  $A = A_1$  and  $B = \sum_{j=2}^c A_j$ . Consider again the decomposition of  $A$  and  $B$  at the  $k$ -th layer into  $A = RP$  and  $B = RQ$ , and let  $\Gamma \in \mathbb{F}^{1 \times w'}$  such that  $\Gamma P = 0$  and  $\Gamma Q \neq 0$ , where  $w' = (d+1)w$  and  $\text{supp}(\Gamma) \leq w+1$ .

The only difference to the proof of Lemma 4.6 is  $Q = [Q_1 Q_2 \dots Q_{w'}]^T$ . Recall from Lemma 4.5 that  $Q_i = B_{(\mathbf{y}_k, \mathbf{a}_i)} = \sum_{j=2}^c A_{j(\mathbf{y}_k, \mathbf{a}_i)}$ , for  $\mathbf{a}_i \in \text{depend}_k(A)$ . Hence,

$$\Gamma Q = \sum_{i=1}^{w'} \gamma_i \left( \sum_{j=2}^c A_{j(\mathbf{y}_k, \mathbf{a}_i)} \right) = \sum_{j=2}^c \sum_{i=1}^{w'} \gamma_i A_{j(\mathbf{y}_k, \mathbf{a}_i)}.$$

By Lemma 2.3,  $\Gamma Q$  can be computed by a sum of  $c-1$  ROABPs, each of width  $w(w+1) \leq 2w^2 = w''$ , because  $\text{supp}(\Gamma) \leq w+1$ . Our definition of  $W_{w,c}$  was chosen such that

$$W_{w'',c-1} = (d+1)(2w'')^{2^{c-2}} = (d+1)(2 \cdot 2w^2)^{2^{c-2}} = (d+1)(2w)^{2^{c-1}} = W_{w,c}.$$

Hence,  $\mathbf{f}_{w,c}(t)$  is a shift that  $\ell_{w'',c-1}$ -concentrates any polynomial that is computed by an ROABP of width  $W_{w'',c-1}$ . By the induction hypothesis, we get that  $\Gamma Q' = \Gamma Q(\mathbf{x} + \mathbf{f}_{w,c}(t))$  is  $(c-1)\ell_{w'',c-1}$ -concentrated, which is same as  $(c-1)\ell_{w,c}$ -concentrated.

Now we can proceed as in the proof of Lemma 4.6 and get that  $(A+B)' = \sum_{j=1}^c A'_j$  has a monomial of support  $< \ell_{w,c} + (c-1)\ell_{w,c} = c\ell_{w,c}$ . ◀

We combine the lemmas similarly as for Theorem 4.7 and obtain our main result for the sum of constantly many ROABPs.

► **Theorem 4.9.** *Given  $n, w, d$ , in time  $(ndw)^{O(c \cdot 2^c \log ndw)}$  one can construct a hitting-set for all  $n$ -variate polynomials of individual degree  $d$ , that can be computed by the sum of  $c$  ROABPs of width  $w$ .*

### 4.2 Concentration in matrix polynomials

As a by-product, we show that low support concentration can be achieved even when we have a sum of matrix polynomials, each computed by an ROABP. For a matrix polynomial  $A(\mathbf{x}) \in F^{w \times w}[\mathbf{x}]$ , an ROABP is defined similar to the standard case. We have layers of nodes  $V_0, V_1, \dots, V_n$  connected by directed edges from  $V_{i-1}$  to  $V_i$ . Here, also  $V_0 = \{v_{0,1}, v_{0,2}, \dots, v_{0,w}\}$  and  $V_n = \{v_{n,1}, v_{n,2}, \dots, v_{n,w}\}$  consist of  $w$  nodes. The polynomial  $A_{i,j}(\mathbf{x})$  at position  $(i, j)$  in  $A(\mathbf{x})$  is the polynomial computed by the standard ROABP with start node  $v_{0,i}$  and end node  $v_{n,j}$ .

Note that Definition 4.2 for  $\ell$ -support concentration can be applied to polynomials over any  $\mathbb{F}$ -algebra.

► **Corollary 4.10.** *Let  $A = A_1 + A_2 + \dots + A_c$ , where each  $A_i \in \mathbb{F}^{w \times w}[\mathbf{x}]$  is an  $n$ -variate matrix polynomials of individual degree  $d$ , each computed by an ROABP of width  $w$ . Let  $\ell_{w,c}$  be defined as in Lemma 4.8.*

*Then  $A(\mathbf{x} + \mathbf{f}_{w^2,c})$  is  $c\ell_{w^2,c}$ -concentrated.*

**Proof.** Let  $\alpha \in \mathbb{F}^{w \times w}$  and consider the dot-product  $\langle \alpha, A_i \rangle \in \mathbb{F}[\mathbf{x}]$ . This polynomial can be computed by an ROABP of width  $w^2$ : we take the ROABP of width  $w$  for  $A_i$  and make  $w$  copies of it, and two new nodes  $s$  and  $t$ . We add the following edges.

- Connect the new start node  $s$  to the  $h$ -th former start node of the  $h$ -th copy of the ROABP by edges of weight one, for all  $1 \leq h \leq w$ .
- Connect the  $j$ -th former end node of the  $h$ -th copy of the ROABP to the new end node  $t$  by an edge of weight  $\alpha_{h,j}$ , for all  $1 \leq h, j \leq w$ .

The resulting ROABP has width  $w^2$  and computes  $\langle \alpha, A_i \rangle$ .

Now consider the polynomial  $\langle \alpha, A \rangle = \langle \alpha, A_1 \rangle + \langle \alpha, A_2 \rangle + \dots + \langle \alpha, A_c \rangle$ . It can be computed by a sum of  $c$  ROABPs, each of width  $w^2$ , for every  $\alpha \in \mathbb{F}^{w \times w}$ . Hence, by Lemma 4.8, the polynomial  $\langle \alpha, A \rangle(\mathbf{x} + \mathbf{f}_{w^2,c})$  is  $c\ell_{w^2,c}$ -concentrated, for every  $\alpha \in \mathbb{F}^{w \times w}$ . By Lemma 4.11 below, it follows that  $A(\mathbf{x} + \mathbf{f}_{w^2,c})$  is  $c\ell_{w^2,c}$ -concentrated. ◀

The following lemma is also of independent interest.

► **Lemma 4.11.** *Let  $A \in \mathbb{F}^{w \times w}[\mathbf{x}]$  be an  $n$ -variate polynomial and  $\mathbf{f}(t)$  be a shift. Then  $A(\mathbf{x} + \mathbf{f}(t))$  is  $\ell$ -concentrated iff  $\forall \alpha \in \mathbb{F}^{w \times w}$ ,  $\langle \alpha, A \rangle(\mathbf{x} + \mathbf{f}(t))$  is  $\ell$ -concentrated.*

**Proof.** Assume that  $A'(\mathbf{x}) = A(\mathbf{x} + \mathbf{f})$  is not  $\ell$ -concentrated. Then there exists a monomial  $\mathbf{x}^b$  such that  $\text{coeff}_{A'}(\mathbf{x}^b) \notin \text{span}_{\mathbb{F}(t)}\{\text{coeff}_{A'}(\mathbf{x}^a) \mid \text{supp}(\mathbf{a}) < \ell\}$ . Hence, there exists an  $\alpha \in \mathbb{F}^{w \times w}$  such that  $\langle \alpha, \text{coeff}_{A'}(\mathbf{x}^a) \rangle = 0$ , for all  $\mathbf{a}$  with  $\text{supp}(\mathbf{a}) < \ell$ , but  $\langle \alpha, A' \rangle \neq 0$ . We thus found an  $\alpha \in \mathbb{F}^{w \times w}$  such that  $\langle \alpha, A \rangle(\mathbf{x} + \mathbf{f}(t))$  is not  $\ell$ -concentrated.

For the other direction, let  $A(\mathbf{x} + \mathbf{f})$  be  $\ell$ -concentrated. Hence, any coefficient  $\text{coeff}_{A'}(\mathbf{x}^a)$  can be written as a linear combination of the small support coefficients,

$$\text{coeff}_{A'}(\mathbf{x}^a) = \sum_{\substack{\mathbf{b} \\ \text{supp}(\mathbf{b}) < \ell}} \gamma_{\mathbf{b}} \text{coeff}_{A'}(\mathbf{x}^b),$$

for some  $\gamma_{\mathbf{b}} \in \mathbb{F}$ . Hence, for any  $\alpha \in \mathbb{F}^{w \times w}$ , we also have

$$\langle \alpha, \text{coeff}_{A'}(\mathbf{x}^a) \rangle = \left\langle \alpha, \sum_{\substack{\mathbf{b} \\ \text{supp}(\mathbf{b}) < \ell}} \gamma_{\mathbf{b}} \text{coeff}_{A'}(\mathbf{x}^b) \right\rangle.$$

That is,  $\langle \alpha, A \rangle(\mathbf{x} + \mathbf{f}(t))$  is  $\ell$ -concentrated. ◀

## 5 Low Support Concentration in ROABPs

Recall that a polynomial  $A(\mathbf{x})$  over an  $\mathbb{F}$ -algebra  $\mathbb{A}$  is called low-support concentrated if its low-support coefficients span all its coefficients. We show an efficient shift which achieves concentration in matrix polynomials computed by ROABPs. We use the quasi-polynomial size hitting-set for ROABPs given by Agrawal et al. [3]. Their hitting-set is based on a *basis isolating weight assignment* which we define next.

Recall that  $M = \{0, 1, \dots, d\}^n$  denotes the set of all exponents of monomials in  $\mathbf{x}$  of individual degree bounded by  $d$ . For a weight function  $w: [n] \rightarrow \mathbb{N}$  and  $\mathbf{a} = (a_1, a_2, \dots, a_n) \in M$ , let the weight of  $\mathbf{a}$  be  $w(\mathbf{a}) = \sum_{i=1}^n w(i)a_i$ . Let  $\mathbb{A}_k$  be a  $k$ -dimensional algebra over field  $\mathbb{F}$ .

► **Definition 5.1.** A weight function  $w: [n] \rightarrow \mathbb{N}$  is called a *basis isolating weight assignment* for a polynomial  $A(\mathbf{x}) \in \mathbb{A}_k[\mathbf{x}]$ , if there exists  $S \subseteq M$  with  $|S| \leq k$  such that

- $\forall \mathbf{a} \neq \mathbf{b} \in S, w(\mathbf{a}) \neq w(\mathbf{b})$  and
- $\forall \mathbf{a} \in \bar{S} := M - S, \text{coeff}_A(\mathbf{x}^{\mathbf{a}}) \in \text{span}_{\mathbb{F}}\{\text{coeff}_A(\mathbf{x}^{\mathbf{b}}) \mid \mathbf{b} \in S \text{ and } w(\mathbf{b}) < w(\mathbf{a})\}$ .

Agrawal et al. [3, Lemma 8] presented a quasi-polynomial time construction of such a weight function for any polynomial  $A(\mathbf{x}) \in \mathbb{F}^{w \times w}[\mathbf{x}]$  computed by an ROABP. The hitting-set is then defined by points  $(t^{w(1)}, t^{w(2)}, \dots, t^{w(n)})$  for  $\text{poly}(n, d, w)^{\log n}$  many  $t$ 's. Our approach now is to use this weight function for a shift of  $A(\mathbf{x})$  by  $(t^{w(i)})_{i=1}^n$ . Let  $A'(\mathbf{x})$  denote the shifted polynomial,

$$A'(\mathbf{x}) = A(\mathbf{x} + t^w) = A(x_1 + t^{w(1)}, x_2 + t^{w(2)}, \dots, x_n + t^{w(n)}).$$

We will prove that  $A'$  has low support concentration.

The coefficients of  $A'$  are linear combinations of coefficients of  $A$ , which are given by the equation

$$\text{coeff}_{A'}(\mathbf{x}^{\mathbf{a}}) = \sum_{\mathbf{b} \in M} \binom{\mathbf{b}}{\mathbf{a}} t^{w(\mathbf{b}-\mathbf{a})} \cdot \text{coeff}_A(\mathbf{x}^{\mathbf{b}}), \quad (21)$$

where  $\binom{\mathbf{b}}{\mathbf{a}} = \prod_{i=1}^n \binom{b_i}{a_i}$  for any  $\mathbf{a}, \mathbf{b} \in \mathbb{N}^n$ .

Equation (21) can be expressed in terms of matrices. Let  $C$  be the coefficient matrix of  $A$ , i.e. the  $M \times [k]$  matrix with the coefficients  $\text{coeff}_A(\mathbf{x}^{\mathbf{a}})$  as rows,

$$C(\mathbf{a}, \cdot) = \text{coeff}_A(\mathbf{x}^{\mathbf{a}})^T.$$

Similarly, let  $C'$  be the  $M \times [k]$  with the coefficients  $\text{coeff}_{A'}(\mathbf{x}^{\mathbf{a}})$  as rows. Let furthermore  $T$  be the  $M \times M$  *transfer matrix* given by

$$T(\mathbf{a}, \mathbf{b}) = \binom{\mathbf{b}}{\mathbf{a}},$$

and  $D$  be the  $M \times M$  diagonal matrix given by

$$D(\mathbf{a}, \mathbf{a}) = t^{w(\mathbf{a})}.$$

The inverse of  $D$  is the diagonal matrix given by  $D^{-1}(\mathbf{a}, \mathbf{a}) = t^{-w(\mathbf{a})}$ . Now equation (21) becomes

$$C' = D^{-1}TDC. \quad (22)$$

As shifting is an invertible operation, the matrix  $T$  is also invertible and  $\text{rank}(C') = \text{rank}(C)$ .

► **Lemma 5.2** (Isolation to concentration). *Let  $A(\mathbf{x})$  be a polynomial over a  $k$ -dimensional algebra  $\mathbb{A}_k$ . Let  $w$  be a basis isolating weight assignment for  $A(\mathbf{x})$ . Then  $A(\mathbf{x} + t^w)$  is  $\ell$ -concentrated, where  $\ell = \lceil \log(k + 1) \rceil$ .*

**Proof.** Let  $A'(\mathbf{x}) = A(\mathbf{x} + t^w)$ . We reconsider equation (22) with respect to the low support monomials: let  $M_\ell = \{\mathbf{a} \in M \mid \text{supp}(\mathbf{a}) < \ell\}$  be the exponents of low support. Then we define matrices

- $C'_\ell$  : the  $M_\ell \times [k]$  submatrix of  $C'$  that contains the coefficients of  $A'$  of support  $< \ell$ ,
- $T_\ell$  : the  $M_\ell \times M$  submatrix of  $T$  restricted to the rows  $\mathbf{a} \in M_\ell$ ,
- $D_\ell$  : the  $M_\ell \times M_\ell$  submatrix of  $D$  restricted to the rows and columns from  $M_\ell$ .

To show that  $A'$  is  $\ell$ -concentrated, we need to prove that  $\text{rank}(C'_\ell) = \text{rank}(C)$ . By equation (22), matrix  $C'_\ell$  can be written as  $C'_\ell = D_\ell^{-1}T_\ell DC$ . Since  $D_\ell$  and  $D_\ell^{-1}$  are diagonal matrices, they have full rank. Hence, it suffices to show that  $\text{rank}(T_\ell DC) = \text{rank}(C)$ .

W.l.o.g. we assume that the order of the rows and columns in all the above matrices that are indexed by  $M$  or  $M_\ell$  is according to increasing weight  $w(\mathbf{a})$  of the indices  $\mathbf{a}$ . The rows with the same weight can be arranged in an arbitrary order.

Now, recall that  $w$  is a basis isolating weight assignment. Hence, there exists a set  $S \subseteq M$  such that the coefficients  $\text{coeff}_A(\mathbf{b})$ , for  $\mathbf{b} \in S$ , span all coefficients  $\text{coeff}_A(\mathbf{a})$ , for  $\mathbf{a} \in M$ . In terms of the coefficient matrix  $C$ , for any  $\mathbf{a} \in M$  we can write

$$C(\mathbf{a}, \cdot) \in \text{span}\{C(\mathbf{b}, \cdot) \mid \mathbf{b} \in S \text{ and } w(\mathbf{b}) < w(\mathbf{a})\}. \tag{23}$$

Let  $S = \{\mathbf{s}_1, \mathbf{s}_2, \dots, \mathbf{s}_{k'}\}$  for some  $k' \leq k$ . Let  $C_0$  be the  $k' \times k$  submatrix of  $C$  whose  $i$ -th row is  $C(\mathbf{s}_i, \cdot)$ , i.e.  $C_0(i, \cdot) = C(\mathbf{s}_i, \cdot)$ . By (23), for every  $\mathbf{a} \in M$ , there is a vector  $\gamma_{\mathbf{a}} = (\gamma_{\mathbf{a},1}, \gamma_{\mathbf{a},2}, \dots, \gamma_{\mathbf{a},k'}) \in \mathbb{F}^{k'}$  such that  $C(\mathbf{a}, \cdot) = \sum_{j=1}^{k'} \gamma_{\mathbf{a},j} C_0(j, \cdot)$ . Let  $\Gamma = (\gamma_{\mathbf{a},j})_{\mathbf{a},j}$  be the  $M \times [k']$  matrix with these vectors as rows. Then we get

$$C = \Gamma C_0.$$

Observe that the  $\mathbf{s}_i$ -th row of  $\Gamma$  is simply  $\mathbf{e}_i$ , the  $i$ -th standard unit vector. By (23), the coefficient  $C(\mathbf{s}_i, \cdot)$  is used to express  $C(\mathbf{a}, \cdot)$  only when  $w(\mathbf{a}) > w(\mathbf{s}_i)$ . Recall that the rows of the matrices indexed by  $M$ , like  $\Gamma$ , are in order of increasing weight of the index. Therefore, when we consider the  $i$ -th column of  $\Gamma$  from top, the entries are all zero down to row  $\mathbf{s}_i$ , where we hit on the one from  $\mathbf{e}_i$ ,

$$\Gamma(\mathbf{s}_i, i) = 1 \text{ and } \forall \mathbf{a} \neq \mathbf{s}_i, w(\mathbf{a}) \leq w(\mathbf{s}_i) \implies \Gamma(\mathbf{a}, i) = 0. \tag{24}$$

Recall that our goal is to show  $\text{rank}(T_\ell DC) = \text{rank}(C)$ . For this, it suffices to show that the  $M_\ell \times k'$  matrix  $R = T_\ell D\Gamma$  has full column rank  $k'$ , because then we have  $\text{rank}(T_\ell DC) = \text{rank}(T_\ell D\Gamma C_0) = \text{rank}(RC_0) = \text{rank}(C_0) = \text{rank}(C)$ .

To show that  $R$  has full column rank  $k'$ , observe that the  $j$ -th column of  $R$  can be written as

$$R(\cdot, j) = \sum_{\mathbf{a} \in M} T_\ell(\cdot, \mathbf{a}) \Gamma(\mathbf{a}, j) t^{w(\mathbf{a})}. \tag{25}$$

By (24), the term with the lowest degree in equation (25) is  $t^{w(\mathbf{s}_j)}$ . By  $\text{lc}(R(\cdot, j))$  we denote the coefficient of the lowest degree term in the polynomial  $R(\cdot, j)$ . Because  $\Gamma(\mathbf{s}_j, j) = 1$ , we have

$$\text{lc}(R(\cdot, j)) = T_\ell(\cdot, \mathbf{s}_j).$$



We define the  $M_\ell \times [k']$  matrix  $R_0$  whose  $j$ -th column is  $\text{lc}(R(\cdot, j))$ , i.e.  $R_0(\cdot, j) = T_\ell(\cdot, \mathbf{s}_j)$ . We will show in Lemma 5.3 below that the columns of matrix  $T_\ell$  indexed by the set  $S$  are linearly independent. Therefore the  $k'$  columns of  $R_0$  are linearly independent.

Hence, there are  $k'$  rows in  $R_0$  such that its restriction to these rows, say  $R'_0$ , is a square matrix with nonzero determinant. Let  $R'$  denote the restriction of  $R$  to the same set of rows. Now observe that the lowest degree term in  $\det(R')$  has coefficient precisely  $\det(R'_0)$ , i.e.,  $\text{lc}(\det(R')) = \det(R'_0)$ . This is because the lowest degree term in  $\det(R')$  has degree  $\sum_{j=1}^{k'} w(\mathbf{s}_j)$ , and this degree can only be obtained when the degree  $w(\mathbf{s}_j)$  term is taken from the  $j$ -th column, for all  $j$ . We conclude that  $\det(R') \neq 0$  and hence  $R$  has full column rank. ◀

It remains to show that the  $k' \leq k$  columns of matrix  $T_\ell$  indexed by the set  $S$  are linearly independent. In fact, we will show that any  $k = 2^\ell - 1$  columns of  $T_\ell$  are independent.

► **Lemma 5.3.** *Let  $T_\ell$  be the  $M_\ell \times M$  matrix with  $T_\ell(\mathbf{a}, \mathbf{b}) = \binom{\mathbf{b}}{\mathbf{a}}$ . Any  $2^\ell - 1$  columns of matrix  $T_\ell$  are linearly independent.*

**Proof.** Let  $S \subseteq M$  now be any set of size  $k = 2^\ell - 1$ . Let  $T_{\ell,k}$  be the  $M_\ell \times S$  submatrix of  $T_\ell$  that consists of the columns indexed by  $S$ . To prove the lemma we will show that for any  $0 \neq \mathbf{v} \in \mathbb{F}^k$  we have  $T_{\ell,k}\mathbf{v} \neq 0$ .

Let  $\mathbf{v} = (v_{\mathbf{a}})_{\mathbf{a} \in S}$ . Define the polynomial  $V(\mathbf{x}) = \sum_{\mathbf{a} \in S} v_{\mathbf{a}} \mathbf{x}^{\mathbf{a}} \in \mathbb{F}[\mathbf{x}]$ . Let  $V'(\mathbf{x})$  be the polynomial where every variable in  $V(\mathbf{x})$  is shifted by one:  $V'(\mathbf{x}) = V(\mathbf{x} + \mathbf{1})$ . From equation (21) we get that for any  $\mathbf{a} \in M_\ell$ ,

$$\text{coeff}_{V'}(\mathbf{x}^{\mathbf{a}}) = \sum_{\mathbf{b} \in S} \binom{\mathbf{b}}{\mathbf{a}} v_{\mathbf{b}} = T_{\ell,k}(\mathbf{a}, \cdot) \mathbf{v}.$$

Hence,  $T_{\ell,k}\mathbf{v}$  gives all the coefficients of  $V'(\mathbf{x})$  of support  $< \ell$ . Now it remains to show that at least one of these coefficients is nonzero. We show this in our next claim about *concentration in sparse polynomials*, which is also of independent interest.

► **Claim 5.4.** *Let  $V(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$  be a non-zero  $n$ -variate polynomial with sparsity bounded by  $2^\ell - 1$ . Then  $V'(\mathbf{x}) = V(\mathbf{x} + \mathbf{1})$  has a nonzero coefficient of support  $< \ell$ .*

We prove the claim by induction on the number of variables  $n$ . For  $n = 1$ , polynomial  $V(\mathbf{x})$  is univariate, i.e. all monomials in  $V(\mathbf{x})$  have support 1. Hence, for  $\ell > 1$  it suffices to show that  $V'(\mathbf{x}) \neq 0$ . But this is equivalent to  $V(\mathbf{x}) \neq 0$ , which holds by assumption. If  $\ell = 1$ , then  $V(\mathbf{x})$  is a univariate polynomial with exactly one monomial, and therefore  $V(\mathbf{x} + \mathbf{1})$  has a nonzero constant part.

Now assume that the claim is true for  $n - 1$  and let  $V(\mathbf{x})$  have  $n$  variables. Let  $\mathbf{x}_{n-1}$  denote the set of first  $n - 1$  variables. Let us write  $V(\mathbf{x}) = \sum_{i=0}^d U_i x_n^i$ , where  $U_i \in \mathbb{F}[\mathbf{x}_{n-1}]$ , for every  $0 \leq i \leq d$ . Let  $U'_i(\mathbf{x}_{n-1}) = U_i(\mathbf{x}_{n-1} + \mathbf{1})$  be the shifted polynomial, for every  $0 \leq i \leq d$ . We consider two cases:

**Case 1:** There is exactly one index  $i \in [0, d]$  for which  $U_i \neq 0$ . Then  $U_i$  has sparsity  $\leq 2^\ell - 1$ .

Because  $U_i$  is an  $(n - 1)$ -variate polynomial,  $U'_i$  has a nonzero coefficient of support  $< \ell$  by inductive hypothesis.

Thus,  $V'(\mathbf{x}) = (x_n + 1)^i U'_i$  also has a nonzero coefficient of support  $< \ell$ .

**Case 2:** There are at least two  $U_i$ 's which are nonzero. Then there is at least one index in  $i \in [0, d]$  such that  $U_i$  has sparsity  $2^{\ell-1} - 1$ . And hence, by the inductive hypothesis,  $U'_i$  has a nonzero coefficient of support  $< \ell - 1$ . Consider the largest index  $j$  such that  $U'_j$



has a nonzero coefficient of support  $< \ell - 1$ . Let the corresponding monomial be  $\mathbf{x}_{n-1}^{\mathbf{a}}$ . Now, as  $V'(\mathbf{x}) = \sum_{i=0}^d U'_i(x_n + 1)^i$ , we have that

$$\text{coeff}_{V'}(\mathbf{x}_{n-1}^{\mathbf{a}} x_n^j) = \sum_{r=j}^d \binom{r}{j} \text{coeff}_{U'_r}(\mathbf{x}_{n-1}^{\mathbf{a}}).$$

By our choice of  $j$  we have  $\text{coeff}_{U'_j}(\mathbf{x}_{n-1}^{\mathbf{a}}) \neq 0$  and  $\text{coeff}_{U'_r}(\mathbf{x}_{n-1}^{\mathbf{a}}) = 0$ , for  $r > j$ . Hence,  $\text{coeff}_{V'}(\mathbf{x}_{n-1}^{\mathbf{a}} x_n^j) \neq 0$ . The monomial  $\mathbf{x}_{n-1}^{\mathbf{a}} x_n^j$  has support  $< \ell$ , which proves our claim and the lemma.  $\blacktriangleleft$

We can use Lemma 5.2 to get concentration in a polynomial computed by an ROABP. Agrawal et al. [3, Lemma 8] constructed a family  $\mathcal{F} = \{\mathbf{f}_1(t), \mathbf{f}_2(t), \dots, \mathbf{f}_N(t)\}$  of  $n$ -tuples such that for any given polynomial  $A(\mathbf{x}) \in \mathbb{F}^{w \times w}[\mathbf{x}]$  computed by an ROABP of width  $w$ , at least one of them is a basis isolating weight assignment and hence, provides  $\log(w^2 + 1)$ -concentration, where  $N = (ndw)^{O(\log n)}$ . The degrees are bounded by  $D = \max\{\deg(f_{i,j}) \mid i \in [N] \text{ and } j \in [n]\} = (ndw)^{O(\log n)}$ . The family  $\mathcal{F}$  can be generated in time  $(ndw)^{O(\log n)}$ .

By Lemma 5.2, we now have an alternative PIT for *one* ROABP because we could simply try all  $\mathbf{f}_i \in \mathcal{F}$  for low support concentration, and we know that at least one will work. However, in Lemmas 4.6 and 4.8 we apply the shift to several ROABPs simultaneously, and we have no guarantee that one of the shifts works for all of them. We solve this problem by combining the  $n$ -tuples in  $\mathcal{F}$  into one single shift that works for every ROABP.

Let  $\mathbf{L}(y, t) \in \mathbb{F}[y, t]^n$  be the Lagrange interpolation of  $\mathcal{F}$ . That is, for all  $j \in [n]$ ,

$$L_j = \sum_{i \in [N]} f_{i,j}(t) \prod_{\substack{i' \in [N] \\ i' \neq i}} \frac{y - \alpha_{i'}}{\alpha_i - \alpha_{i'}},$$

where  $\alpha_i$  is an arbitrary unique field element associated with  $i$ , for all  $i \in [N]$ . (Recall that we assume that the field  $\mathbb{F}$  is large enough that these elements exist.) Note that  $L_j|_{y=\alpha_i} = f_{i,j}$ . Thus,  $\mathbf{L}|_{y=\alpha_i} = \mathbf{f}_i$ . Also,  $\deg_y(L_j) = N - 1$  and  $\deg_t(L_j) \leq D$ .

**► Lemma 5.5.** *Let  $A(\mathbf{x})$  be a  $n$ -variate polynomial over a  $k$ -dimensional  $\mathbb{F}$ -algebra  $\mathbb{A}_k$  and  $\mathcal{F}$  be a family of  $n$ -tuples, such that there exists an  $\mathbf{f} \in \mathcal{F}$  such that  $A'(\mathbf{x}, t) = A(\mathbf{x} + \mathbf{f}) \in \mathbb{A}_k(t)[\mathbf{x}]$  is  $\ell$ -concentrated. Then,  $A''(\mathbf{x}, y, t) = A(\mathbf{x} + \mathbf{L}) \in \mathbb{A}_k(y, t)[\mathbf{x}]$  is  $\ell$ -concentrated.*

**Proof.** Let  $\text{rank}_{\mathbb{F}}\{\text{coeff}_A(\mathbf{x}^{\mathbf{a}}) \mid \mathbf{a} \in M\} = k'$ , for some  $k' \leq k$ , and  $M_{\ell} = \{\mathbf{a} \in M \mid \text{supp}(\mathbf{a}) < \ell\}$ . We need to show that  $\text{rank}_{\mathbb{F}(y,t)}\{\text{coeff}_{A''}(\mathbf{x}^{\mathbf{a}}) \mid \mathbf{a} \in M_{\ell}\} = k'$ .

Since  $A'(\mathbf{x})$  is  $\ell$ -concentrated, we have that  $\text{rank}_{\mathbb{F}(t)}\{\text{coeff}_{A'}(\mathbf{x}^{\mathbf{a}}) \mid \mathbf{a} \in M_{\ell}\} = k'$ . Recall that  $A'(\mathbf{x})$  is an evaluation of  $A''$  at  $y = \alpha_i$ , i.e.  $A'(\mathbf{x}, t) = A''(\mathbf{x}, \alpha_i, t)$ . Thus, for all  $\mathbf{a} \in M$  we have  $\text{coeff}_{A'}(\mathbf{x}^{\mathbf{a}}) = \text{coeff}_{A''}(\mathbf{x}^{\mathbf{a}})|_{y=\alpha_i}$ .

Let  $C \in \mathbb{F}[t]^{k \times |M_{\ell}|}$  be the matrix whose columns are  $\text{coeff}_{A'}(\mathbf{x}^{\mathbf{a}})$ , for  $\mathbf{a} \in M_{\ell}$ . Let similarly  $C' \in \mathbb{F}[y, t]^{k \times |M_{\ell}|}$  be the matrix whose columns are  $\text{coeff}_{A''}(\mathbf{x}^{\mathbf{a}})$ , for  $\mathbf{a} \in M_{\ell}$ . Then we have  $C = C'|_{y=\alpha_i}$ .

As  $\text{rank}_{\mathbb{F}(t)}(C) = k'$ , there are  $k'$  rows in  $C$ , say indexed by  $R$ , such that  $\det(C(R, \cdot)) \neq 0$ . Because  $\det(C(R, \cdot)) = \det(C'(R, \cdot))|_{y=\alpha_i}$ , it follows that  $\det(C'(R, \cdot)) \neq 0$ . Hence, we have  $\text{rank}_{\mathbb{F}(y,t)}(C') = k'$ .  $\blacktriangleleft$

Using the Lagrange interpolation, we can construct a single shift, which works for all ROABPs of width  $\leq w$ .

**► Theorem 5.6.** *Given  $n, d, w$ , in time  $(ndw)^{O(\log n)}$  one can compute a polynomial  $\mathbf{f}(t) \in \mathbb{F}[t]^n$  of degree  $(ndw)^{O(\log n)}$  such that for any  $n$ -variate polynomial  $A(\mathbf{x}) \in \mathbb{F}^{w \times w}[\mathbf{x}]$  (or*

$\mathbb{F}^{1 \times w}[\mathbf{x}]$ , or  $\mathbb{F}[\mathbf{x}]$ ) of individual degree  $d$  that can be computed by an ROABP of width  $w$ , the polynomial  $A(\mathbf{x} + \mathbf{f}(t))$  is  $\log(w^2 + 1)$ -concentrated.

**Proof.** Recall that for any polynomial  $A(\mathbf{x}) \in \mathbb{F}^{w \times w}[\mathbf{x}]$  computed by an ROABP, at least one tuple in the family  $\{\mathbf{f}_1, \mathbf{f}_2, \dots, \mathbf{f}_N\}$  obtained from [3, Lemma 8], gives  $\log(w^2 + 1)$ -concentration. By Lemma 5.5, the Lagrange interpolation  $\mathbf{L}(y, t)$  of  $\{\mathbf{f}_1, \mathbf{f}_2, \dots, \mathbf{f}_N\}$  has  $y$ - and  $t$ -degrees  $(ndw)^{O(\log n)}$ . After shifting an  $n$ -variate polynomial of individual degree  $d$  by  $\mathbf{L}(y, t)$ , its coefficients will be polynomials in  $y$  and  $t$ , with degree  $d' = dn(ndw)^{O(\log n)}$ . Consider the determinant polynomial  $\det(C'(R, \cdot))$  from Lemma 5.5. As the set of coefficients of polynomial  $A(\mathbf{x})$  have rank bounded by  $w^2$ ,  $\det(C'(R, \cdot))$  has degree bounded by  $d'' = w^2 d'$ .

Note that when we replace  $y$  by  $t^{d''+1}$ , this will not affect the non-zerosness of the determinant, and hence, the concentration is preserved. Thus,  $\mathbf{f} = \mathbf{L}(t^{d''+1}, t)$  is an  $n$ -tuple of univariate polynomials in  $t$  that fulfills the claim of the theorem.

Now, consider the case when the ROABP computes a polynomial  $A(\mathbf{x}) \in \mathbb{F}^{1 \times w}[\mathbf{x}]$ . It is easy to see that there exist  $S \in \mathbb{F}^{1 \times w}$  and  $B \in \mathbb{F}^{w \times w}[\mathbf{x}]$  computed by a width- $w$  ROABP such that  $A = SB$ . We know that  $B(\mathbf{x} + \mathbf{f}(t))$  has  $\log(w^2 + 1)$ -concentration. As multiplying by  $S$  is a linear operation, one can argue as in the proof of Lemma 4.11 that any linear dependence among coefficients of  $B(\mathbf{x} + \mathbf{f}(t))$  also holds among coefficients of  $A(\mathbf{x} + \mathbf{f}(t))$ . Hence,  $A(\mathbf{x} + \mathbf{f}(t))$  has  $\log(w^2 + 1)$ -concentration. A similar argument would work when  $A(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$ , by writing  $A = SBT$ , for some  $S \in \mathbb{F}^{1 \times w}$  and  $T \in \mathbb{F}^{w \times 1}$ . ◀

## 6 Discussion

The first question is whether one can make the time complexity for PIT for the sum of  $c$  ROABPs proportional to  $w^{O(c)}$  instead of  $w^{O(2^c)}$ . This blow up happens because, when we want to combine  $w + 1$  partial derivative polynomials given by ROABPs of width  $w$ , we get an ROABP of width  $O(w^2)$ . There are examples where this bound seems tight. So, a new property of sum of ROABPs needs to be discovered.

It also needs to be investigated if these ideas can be generalized to work for sum of more than constantly many ROABPs, or depth-3 multilinear circuits.

As mentioned in the introduction, the idea for equivalence of two ROABPs was inspired from the equivalence of two read once boolean branching programs (OBDD). It would be interesting to know if there are concrete connections between arithmetic and boolean branching programs. In particular, can ideas from identity testing of an ROABP be applied to construct pseudo-randomness for OBDD. E.g. the less investigated model, XOR of constantly many OBDDs can be checked for unsatisfiability by modifying our techniques.

**Acknowledgements.** We thank Manindra Agrawal, Chandan Saha and Vineet Nair for very useful discussions and constant encouragement. The work was initiated when TT was visiting CSE, IIT Kanpur. Part of the work was done during Dagstuhl Seminar 14391 on Algebra in Computational Complexity 2014. We thank anonymous referees for the useful suggestions.

---

## References

- 1 Leonard M. Adleman and Hendrik W. Lenstra. Finding irreducible polynomials over finite fields. In *Proceedings of the Eighteenth Annual ACM Symposium on Theory of Computing*, STOC'86, pages 350–355, New York, NY, USA, 1986. ACM.
- 2 Manindra Agrawal. Proving lower bounds via pseudo-random generators. In *FSTTCS*, volume 3821 of *Lecture Notes in Computer Science*, pages 92–105, 2005.

- 3 Manindra Agrawal, Rohit Gurjar, Arpita Korwar, and Nitin Saxena. Hitting-sets for ROABP and sum of set-multilinear circuits. *Electronic Colloquium on Computational Complexity (ECCC)*, 21:85, 2014. (to appear in SICOMP, 2015).
- 4 Manindra Agrawal, Chandan Saha, and Nitin Saxena. Quasi-polynomial hitting-set for set-depth- formulas. In *STOC*, pages 321–330, 2013.
- 5 Michael Ben-Or and Prason Tiwari. A deterministic algorithm for sparse multivariate polynomial interpolation. In *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, STOC'88, pages 301–309, New York, NY, USA, 1988. ACM.
- 6 Rafael Mendes de Oliveira, Amir Shpilka, and Ben Lee Volk. Subexponential size hitting sets for bounded depth multilinear formulas. *Electronic Colloquium on Computational Complexity (ECCC)*, 21:157, 2014. (to appear in CCC'15).
- 7 Zeev Dvir and Amir Shpilka. Locally decodable codes with two queries and polynomial identity testing for depth 3 circuits. *SIAM J. Comput.*, 36(5):1404–1434, 2007.
- 8 Michael A. Forbes. *Polynomial Identity Testing of Read-Once Oblivious Algebraic Branching Programs*. PhD thesis, MIT, 2014.
- 9 Michael A. Forbes, Ramprasad Saptharishi, and Amir Shpilka. Hitting sets for multilinear read-once algebraic branching programs, in any order. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 – June 03, 2014*, pages 867–875, 2014.
- 10 Michael A. Forbes and Amir Shpilka. On identity testing of tensors, low-rank recovery and compressed sensing. In *STOC*, pages 163–172, 2012.
- 11 Michael A. Forbes and Amir Shpilka. Quasipolynomial-time identity testing of non-commutative and read-once oblivious algebraic branching programs. *CoRR*, abs/1209.2408, 2012.
- 12 Michael A. Forbes and Amir Shpilka. Quasipolynomial-time identity testing of non-commutative and read-once oblivious algebraic branching programs. In *FOCS*, pages 243–252, 2013.
- 13 Ankit Gupta, Pritish Kamath, Neeraj Kayal, and Ramprasad Saptharishi. Arithmetic circuits: A chasm at depth three. *FOCS*, pages 578–587, 2013.
- 14 Maurice J. Jansen, Youming Qiao, and Jayalal Sarma. Deterministic identity testing of read-once algebraic branching programs. *Electronic Colloquium on Computational Complexity (ECCC)*, 17:84, 2010.
- 15 Valentine Kabanets and Russell Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *STOC*, pages 355–364, 2003.
- 16 Zohar Shay Karnin and Amir Shpilka. Black box polynomial identity testing of generalized depth-3 arithmetic circuits with bounded top fan-in. *Combinatorica*, 31(3):333–364, 2011.
- 17 Neeraj Kayal and Shubhangi Saraf. Blackbox polynomial identity testing for depth 3 circuits. In *FOCS*, pages 198–207, 2009.
- 18 Neeraj Kayal and Nitin Saxena. Polynomial identity testing for depth 3 circuits. *Computational Complexity*, 16(2):115–138, 2007.
- 19 Adam Klivans and Daniel A. Spielman. Randomness efficient identity testing of multivariate polynomials. In *STOC*, pages 216–223, 2001.
- 20 Vineet Nair and Chandan Saha. Personal communication, 2014.
- 21 Noam Nisan. Lower bounds for non-commutative computation (extended abstract). In *Proceedings of the 23rd ACM Symposium on Theory of Computing*, ACM Press, pages 410–418, 1991.
- 22 Ran Raz and Amir Shpilka. Deterministic polynomial identity testing in non-commutative models. *Computational Complexity*, 14(1):1–19, 2005.
- 23 Ran Raz and Amir Yehudayoff. Lower bounds and separations for constant depth multilinear circuits. *Computational Complexity*, 18(2):171–207, 2009.

- 24 Petr Savický and Ingo Wegener. Efficient algorithms for the transformation between different types of binary decision diagrams. *Acta Informatica*, 34(4):245–256, 1997.
- 25 Nitin Saxena. Progress on polynomial identity testing. *Bulletin of the EATCS*, 99:49–79, 2009.
- 26 Nitin Saxena. Progress on polynomial identity testing-II. In *Perspectives in Computational Complexity*, pages 131–146. Birkhäuser Basel, 2014.
- 27 Nitin Saxena and Comandur Seshadhri. An almost optimal rank bound for depth-3 identities. *SIAM J. Comput.*, 40(1):200–224, 2011.
- 28 Nitin Saxena and Comandur Seshadhri. Blackbox identity testing for bounded top-fanin depth-3 circuits: The field doesn't matter. *SIAM J. Comput.*, 41(5):1285–1298, 2012.
- 29 Jacob T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *J. ACM*, 27(4):701–717, October 1980.
- 30 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.

# Kolmogorov Width of Discrete Linear Spaces: an Approach to Matrix Rigidity

Alex Samorodnitsky<sup>\*1</sup>, Ilya Shkredov<sup>†2</sup>, and Sergey Yekhanin<sup>3</sup>

1 Hebrew University  
Israel

salex@cs.huji.ac.il

2 Steklov Mathematical Institute and IITP RAS  
Russia

ilya.shkredov@gmail.com

3 Microsoft Research  
USA

yekhanin@microsoft.com

---

## Abstract

---

A square matrix  $V$  is called rigid if every matrix  $V'$  obtained by altering a small number of entries of  $V$  has sufficiently high rank. While random matrices are rigid with high probability, no explicit constructions of rigid matrices are known to date. Obtaining such explicit matrices would have major implications in computational complexity theory. One approach to establishing rigidity of a matrix  $V$  is to come up with a property that is satisfied by any collection of vectors arising from a low-dimensional space, but is not satisfied by the rows of  $V$  even after alterations. In this paper we propose such a candidate property that has the potential of establishing rigidity of combinatorial design matrices over the field  $\mathbb{F}_2$ .

Stated informally, we conjecture that under a suitable embedding of  $\mathbb{F}_2^n$  into  $\mathbb{R}^n$ , vectors arising from a low dimensional  $\mathbb{F}_2$ -linear space always have somewhat small Kolmogorov width, i.e., admit a non-trivial simultaneous approximation by a low dimensional Euclidean space. This implies rigidity of combinatorial designs, as their rows do not admit such an approximation even after alterations. Our main technical contribution is a collection of results establishing weaker forms and special cases of the conjecture above.

**1998 ACM Subject Classification** F.1.2 Modes of Computation, E.4 Coding and Information Theory

**Keywords and phrases** Matrix rigidity, linear codes, Kolmogorov width

**Digital Object Identifier** 10.4230/LIPIcs.CCC.2015.347

## 1 Introduction

The notion of matrix rigidity was introduced by Leslie Valiant in 1977 [23]. In this paper we say that an  $n \times n$  matrix  $A$  defined over a field is  $(r, d)$ -rigid, if it is not possible to reduce the rank of  $A$  below  $r$  by arbitrarily altering each row of  $A$  in up to  $d$  coordinates. Explicit rigid matrices are known to imply lower bounds for computational complexity of explicit functions.

---

\* Alex Samorodnitsky was supported by BSF and ISF grants.

† Ilya Shkredov was supported by Russian Scientific Foundation grant RSF 14-11-00433.



The most prominent reduction of this nature is due to Valiant [23] who showed that for each  $(\Omega(n), n^\epsilon)$ -rigid matrix  $A \in \mathbb{F}^{n \times n}$  the linear transformation induced by  $A$  cannot be computed by a linear circuit that simultaneously has size  $O(n)$  and depth  $O(\log n)$ . Two other reductions that call for explicit  $(r, d)$ -rigid matrices with a sub-linear value of the remaining rank  $r$ , are given in [15, 18]. Reductions above naturally lead to the challenge of constructing rigid matrices explicitly. After more than three decades of efforts, however, this challenge remains elusive [13].

None of the existing techniques for constructing rigid matrices [11, 12, 3, 17, 5, 2] surpasses the basic *untouched minor* argument of [19] that amounts to taking a matrix where every minor has full rank, and using the bound from the Zarankiewicz problem [9, p. 25] to show that after up to  $d$  arbitrary changes per row there remains a somewhat large minor that has not been touched. Quantitatively, this yields explicit

$$\left( r, \Omega\left(\frac{n}{r} \log \frac{n}{r}\right) \right)$$

rigid matrices over fields of size  $\Omega(n)$ , when  $\log^2 n \leq r \leq n/2$ . Similar parameters are known to be attainable over small finite fields [6]. Rigidity parameters above are vastly weaker the parameters of random matrices. In particular, it is not hard to show that a random matrix over any field is at least

$$\left(\frac{n}{2}, \Omega(n)\right)$$

rigid with a very high probability.

### 1.1 Combinatorial designs

A family  $\mathcal{F}$  of  $w$ -subsets of a universe of size  $n$  is called an  $(n, w, \lambda)$  design if every pair of distinct elements of  $[n]$  belongs to exactly  $\lambda$  sets in  $\mathcal{F}$ . A combinatorial design is symmetric if  $|\mathcal{F}| = n$ . Geometric designs are a well studied class of symmetric combinatorial designs. A geometric design is defined by the incidence relation between points and hyperplanes in an  $m$ -dimensional projective space  $\mathbb{P}\mathbb{G}(m + 1, q)$  over the finite field  $\mathbb{F}_q$ . Such a relation yields  $(n, w, \lambda)$  symmetric designs, where

$$n = \frac{q^{m+1} - 1}{q - 1} \quad w = \frac{q^m - 1}{q - 1} \quad \lambda = \frac{q^{m-1} - 1}{q - 1}. \tag{1}$$

With a slight abuse of notation we write  $G_{m,q}$  or just  $G_m$  to denote both geometric designs and their incidence matrices.

In his original paper [23, Problem 4] Valiant proposed matrices  $G_2$  defined above as natural candidates for  $(\Omega(n), n^\epsilon)$ -rigidity over the field  $\mathbb{F}_2$ . Taken literally, this conjecture is not true as some matrices  $G_2$  have low rank over  $\mathbb{F}_2$ . In fact, the rank of geometric designs is a well studied quantity in design theory. Let  $\text{rank}_p$  denote matrix rank over the field  $\mathbb{F}_p$ . We have [20]:

$$\text{rank}_p G_{m,q} = \begin{cases} n & \text{if } q \neq p^e, \quad w + (n - 1)\lambda \neq 0 \pmod p; \\ n - 1 & \text{if } q \neq p^e, \quad w + (n - 1)\lambda = 0 \pmod p; \\ \binom{p+m-1}{m}^e & \text{if } q = p^e. \end{cases} \tag{2}$$

Thus in some cases the rank of geometric designs turns out to be surprisingly low, e.g., when  $\text{char } \mathbb{F}_q = 2$ , for fixed  $m$  and growing  $q$  we have

$$\text{rank}_2 G_{m,q} = O\left(n^{\frac{\log_2(m+1)}{m}}\right). \tag{3}$$

Identity (3) implies that any proof of  $(r, d)$ -rigidity of matrices  $G_m$  with  $r = \Omega(n)$  cannot just rely on the combinatorial structure of these matrices as this structure does not seem to change much with the characteristic of the field underlying the projective space. Thus any rigidity proof that relies solely on the design properties of  $G_m$  (and thus applies to all designs with the parameters of geometric designs) has to be aiming at the regime of polynomially low remaining rank  $r = O(n^\delta)$ . In Section 1.3 we outline our approach to proving a result like this.

### 1.2 Hamada’s conjecture

In what follows let  $V_m$  denote an incidence matrix (or the set of rows of an incidence matrix) of a combinatorial  $(n, w, \lambda)$  design that has the parameters of the geometric design  $G_{m,q}$ . Clearly, any proof of  $(r, d)$ -rigidity of  $V_m$  has to imply that matrices  $V_m$  have rank at least  $r$  when no alterations are allowed. Bounding the rank of matrices  $V_m$  over finite fields has received some attention in design theory.

It is not hard to show that when  $q \neq p^e$  we have,  $\text{rank}_p V_m \geq n - 1$ . A conjecture [7] due to Noboru Hamada from 1973, asserts that when  $q = p^e$ , geometric designs  $G_{m,q}$  have the lowest possible  $\mathbb{F}_p$ -rank among all designs  $V_m$  with the same parameters. Relatively little is known about the validity of Hamada’s conjecture [10, Section 4]. (A stronger version of Hamada’s conjecture that asserts that every design  $V_m$  whose  $\mathbb{F}_p$ -rank equals that of  $G_{m,q}$  has to be isomorphic to  $G_{m,q}$  is known to be false [10].)

One easier natural question to ask that fits well with our approach to rigidity is whether one can prove any non-trivial lower bounds on the rank of design matrices  $V_m$ . We are particularly interested in the asymptotic setting of fixed  $m$  and growing  $q$ . Hamada’s conjecture and identity (2) suggest that

$$\text{rank}_p V_m \geq \Omega \left( n^{\frac{\log_p \binom{p+m-1}{m}}{m}} \right). \tag{4}$$

The trivial lower bound is  $\text{rank}_p V_m \geq n^{\frac{1}{m}}$ . We are not aware of any better bound.

### 1.3 Our approach

In order to establish rigidity of matrices  $V_m$  over the field  $\mathbb{F}_2$  we propose a certain property that is not satisfied by the rows of  $V_m$  even after alterations, yet that we conjecture to hold for any collection of vectors arising from a low-dimensional  $\mathbb{F}_2$ -linear space.

As a first step of our argument we consider a natural embedding of the space  $\mathbb{F}_2^n$  into  $\mathbb{R}^n$ . We treat elements of  $\mathbb{F}_2^n$  as real  $\{0, 1\}$ -vectors and normalize them to have  $L_2$  norm one. Thus a non-zero  $x \in \mathbb{F}_2^n$  gets mapped to  $\frac{x}{\|x\|}$ . In what follows we assume that this embedding is implied and treat vectors in  $\mathbb{F}_2^n$  as real vectors.

Next for sets  $X \subseteq \mathbb{R}^n$  we consider the quantity  $A_r(X)$  that we call the approximability measure.  $A_r(X)$  is defined to be the maximum over all  $r$ -dimensional Euclidian linear spaces  $W$  of the square of the smallest projection of a vector from  $X$  onto  $W$ . Thus sets with large value of  $A_r$  are precisely those that are well approximated by some  $r$ -dimensional Euclidian linear space, i.e., the ones that have small Kolmogorov width.

Now let  $m = \frac{1}{\epsilon}$ . We argue that for all values of  $r = \omega(n^{1-\epsilon})$ , the approximability measure  $A_r(V_m) \approx A_r(\mathbb{F}_2^n)$ . Thus for sufficiently large  $r$ , approximating rows of an incidence matrix of a combinatorial design is no easier than approximating all of the Hamming space. The claim remains true even if we allow rows of  $V_m$  be altered in up to  $O(n^{1-2\epsilon})$  coordinates.

Finally, we conjecture that for any  $\mathbb{F}_2$ -linear space  $L$ ,  $\dim L \leq n^{2\epsilon+\delta}$ , and some  $r = \omega(n^{1-\epsilon})$ , the approximability measure  $A_r(L) \geq (1 + \alpha)A_r(\mathbb{F}_2^n)$ , for some positive  $\alpha$ . In other words, we conjecture that low dimensional  $\mathbb{F}_2$ -linear spaces keep some tiny amount of resemblance to Euclidian linear spaces after the embedding, and can be approximated better than all of the Hamming cube. It is easy to see that this conjecture implies  $(n^{2\epsilon+\delta}, n^{1-2\epsilon})$ -rigidity of matrices  $V_m$ , since if one of these matrices had low rank after alterations, its rows would belong to a low dimensional  $\mathbb{F}_2$ -linear space, and thus admit a non-trivial Euclidian approximation.

We measure our progress towards the conjecture by looking at the largest value of dimension  $k$  for that we are indeed able to prove that all  $k$ -dimensional  $\mathbb{F}_2$ -linear spaces  $L$  satisfy  $A_r(L) \geq (1 + \alpha)A_r(\mathbb{F}_2^n)$ , for some  $r = \omega(n^{1-\epsilon})$ . Currently, our main Theorem 5.5 gives this for all  $k = o(n^\epsilon \log n)$ . Apart from this result, we also establish the conjecture for a certain restricted class of linear spaces called cut-spaces. While substantial further progress is needed to establish rigidity of matrices  $V_m$ , our current results (Corollary 5.6) already suffice to get the bound

$$\text{rank}_p V_m \geq \Omega\left(n^{\frac{1}{m}} \log_2 n\right), \quad (5)$$

for all values of  $p$ , a results that seems to be new.

From the technical viewpoint our main contribution is a new relation between discrete ( $\mathbb{F}_2$ ) linear spaces and Euclidian linear spaces, yielding some insight into combinatorics of low weight codewords in linear codes.

## 1.4 Organization

In Section 3 we formally introduce the approximability measure  $A_r$ . We argue that for sufficiently large values of dimension  $r$ , we have  $A_r(V_m) \approx A_r(\mathbb{F}_2^n)$ . We establish a similar result for perturbed matrices  $V_m$ . Next, we introduce our main conjecture stating that low-dimensional  $\mathbb{F}_2$ -linear spaces  $L$  always have a somewhat large value of  $A_r$ . We show how this conjecture implies rigidity of matrices  $V_m$ .

In Sections 4 through 6 we prove our main results regarding approximability of low-dimensional  $\mathbb{F}_2$ -linear spaces  $L$ . In Section 4 we deal with low-dimensional approximations and obtain a bound for  $A_1(L)$ . In Section 5 we deal with high dimensional approximations and state the implications of our results for the Hamada's conjecture. All our results in Sections 4 and 5 apply not just to  $\mathbb{F}_2$ -linear spaces but to all families of vectors that have bounded triangular rank [14].

In Section 6 we establish our main approximability conjecture for a certain class a linear spaces called cut-spaces and give a simpler proof of a slightly weaker version of the results from Section 5. Our constructions of approximating real spaces use low weight vectors in the dual space of  $L$ . Finally, in Section 7 we discuss the relation of our approach to matrix rigidity to the natural proofs lower bounds barrier of Razborov and Rudich [16].

## 2 Notation

We use the following standard mathematical notation:

- $\|\cdot\|$  denotes the Euclidian norm;
- For an integer  $n$ ,  $[n] = \{1, \dots, n\}$ ;
- For a vector  $\mathbf{v}$ , the set of non-zero coordinates of  $\mathbf{v}$  is denoted  $\text{supp}(\mathbf{v})$ ;
- We write  $f(n) \approx g(n)$ , if  $f(n) = g(n)(1 + o(1))$ . We adopt the same agreement for  $\lesssim, \gtrsim$ .



### 3 The conjecture

We now introduce our approximability measure  $A_r$ . Following that, in Section 3.2 we argue that for sufficiently large values of dimension  $r$ , collections of rows of incidence matrices of combinatorial designs have essentially the smallest possible value of  $A_r$  even after alterations. Finally, in Section 3.3 we introduce our main conjecture stating that low-dimensional  $\mathbb{F}_2$ -linear spaces always have a somewhat large value of  $A_r$ . We show how this conjecture implies rigidity of incidence matrices of combinatorial designs.

#### 3.1 The approximability measure

We consider a natural embedding of  $\mathbb{F}_2^n$  into  $\mathbb{R}^n$ . We treat elements of  $\mathbb{F}_2^n$  as real  $\{0, 1\}$ -vectors and normalize them to have  $L_2$  norm one. Thus a non-zero  $\mathbf{v} \in \mathbb{F}_2^n$  gets mapped to  $\frac{\mathbf{v}}{\|\mathbf{v}\|}$ . Zero is mapped to zero. In what follows we assume that this embedding is implied and treat vectors in  $\mathbb{F}_2^n$  as real vectors. Let  $V$  be an arbitrary subset of  $\mathbb{F}_2^n$ . Our approach is centered around the following approximability measure

$$A_r(V) = \max_{\dim W \leq r} \min_{\mathbf{v} \in V} \|\text{Pr}_W(\mathbf{v})\|^2, \tag{6}$$

where the maximum is taken over all linear spaces  $W \in \mathbb{R}^n, \dim W = r$ , and minimum is taken over the non-zero elements of  $V$ . Observe that our notion of approximability measure is equivalent to the classical concept of Kolmogorov width  $K_r(V) = \sqrt{1 - A_r(V)}$ , also known as "poperechnik" of a family of vectors. See [21, 22].

We remark the importance of the normalization step in formula (6). In essence normalizing elements of  $V$  pushes all high weight vectors in  $V$  to the center of the positive orthant in  $\mathbb{R}^n$ , and makes  $A_r(V)$  governed by the distribution of low weight vectors in  $V$  as needed for our approach. We now derive a formula for approximability measure of the whole Boolean cube.

► **Lemma 3.1.** *Let  $r = o(n)$  be arbitrary. We have*

$$A_r(\mathbb{F}_2^n) \approx \frac{r}{n}. \tag{7}$$

**Proof.** Let  $\mathbf{e}_i, i \in [n]$  denote the  $i$ -th unit vector. First we show that

$$A_r(\mathbb{F}_2^n) \leq A_r(\{\mathbf{e}_1, \dots, \mathbf{e}_n\}) \leq \frac{r}{n}. \tag{8}$$

Let  $W$  be an arbitrary  $r$ -dimensional linear space with an orthonormal basis  $\{\mathbf{w}_1, \dots, \mathbf{w}_r\}$ . Consider an  $n \times r$  matrix  $M$ , where  $M_{ij} = (\mathbf{e}_i, \mathbf{w}_j)^2$ . Clearly, the sum of values in  $M$  is equal to  $r$ . Thus for some  $i \in [n]$  we have  $\sum_j (\mathbf{e}_i, \mathbf{w}_j)^2 \leq \frac{r}{n}$  and (8) follows.

We now exhibit a space  $W$  such that for all non-zero binary vectors  $\mathbf{v}$ ,  $\|\text{Pr}_W(\mathbf{v})\|^2 \gtrsim \frac{r}{n}$ . The space  $W$  is spanned by  $r$  unit vectors  $\{\mathbf{w}_i\}$ . These vectors have disjoint supports that partition  $[n]$ . Every support is of size  $\lceil \frac{n}{r} \rceil$  or  $\lfloor \frac{n}{r} \rfloor$ . Each vector  $\mathbf{w}_i$  is constant on its support. Let  $\mathbf{v}$  be an arbitrary vector of weight  $w$ . Assume that the support of  $\mathbf{v}$  intersects the supports of  $t$  different vectors  $\{\mathbf{w}_i\}$ , namely,  $\mathbf{w}_{i_1}, \dots, \mathbf{w}_{i_t}$ . Clearly,  $t \leq w$ . For  $j \leq t$ , let  $a_j = |\text{supp}(\mathbf{v}) \cap \text{supp}(\mathbf{w}_{i_j})|$ . We have  $\sum_{j=1}^t a_j = w$ . We also have

$$\sum_{j=1}^t (\mathbf{v}, \mathbf{w}_{i_j})^2 \geq \sum_{j=1}^t \frac{a_j^2 r}{(n+r)w} = \frac{r}{(n+r)w} \sum_{j=1}^t a_j^2 \geq \frac{r}{w(n+r)} \left(\frac{w}{t}\right)^2 t \gtrsim \frac{r}{n}.$$

This concludes the proof. ◀

### 3.2 Inapproximability of combinatorial designs

In this Section we argue that approximating rows of a combinatorial design by a high-dimensional real space is as hard as approximating all of the Boolean cube. We also establish a robust version of this result.

► **Lemma 3.2.** *Let  $V \subseteq \mathbb{F}_2^n$ ,  $|V| = n$ . Let  $B$  be the  $n \times n$  real matrix, where the rows of  $B$  are the normalized elements of  $V$ . Let  $\lambda_1 \geq \dots \geq \lambda_n \geq 0$  be the eigenvalues of  $BB^t$ ; then for all  $r$ ,*

$$A_r(V) \leq \frac{1}{n} \sum_{i \leq r} \lambda_i. \quad (9)$$

**Proof.** This is a simple corollary of a result in [8]. A special case of this result states that for all  $r$

$$\frac{1}{n} \sum_{i \leq r} \lambda_i = \max_{\dim W \leq r} \mathbb{E}_{\mathbf{v} \in V} \|\text{Pr}_W(\mathbf{v})\|^2, \quad (10)$$

where the expectation is taken with respect to the uniform distribution on  $V$ . Since the RHS of this equality is at least as large as  $A_r(V)$ , the claim of the Lemma follows. ◀

Let  $V_m$  be an incidence matrix of a combinatorial  $(n, w, \lambda)$  design with the parameters (1) of the geometric design  $G_{m,q}$  for some value of  $q$ . Let  $m = \frac{1}{\epsilon}$ . We assume that  $m$  is fixed and  $q$  grows to infinity. Thus  $w \approx n^{1-\epsilon}$  and  $\lambda \approx n^{1-2\epsilon}$ . Lemma 3.2 yields

► **Corollary 3.3.** *With the notation above, we have*

$$A_r(V_{1/\epsilon}) \lesssim \frac{n^{1-\epsilon} + r}{n}. \quad (11)$$

**Proof.** Let  $B$  be the  $n \times n$  matrix, where the rows of  $B$  are the normalized elements of  $V_m$ . Clearly,  $B = \frac{1}{\sqrt{w}} V_m$ . We have

$$BB^t = \frac{w - \lambda}{w} I + \frac{\lambda}{w} J,$$

where  $I$  denotes the identity matrix and  $J$  denotes the all-ones matrix. It is not hard to see that the eigenvalues of  $BB^t$  are given by

$$\lambda_1 \approx n^{1-\epsilon} \quad \text{and} \quad \lambda_2 = \dots = \lambda_n \approx 1.$$

An application of Lemma 3.2 completes the proof. ◀

Combining (7) and (11), we conclude that for  $r = \omega(n^{1-\epsilon})$  and  $r = o(n)$ ,

$$A_r(V_{1/\epsilon}) \approx A_r(\mathbb{F}_2^n) \approx \frac{r}{n}. \quad (12)$$

Observe that identity (10) and the eigenvalue computation above can be used to show that matrices  $V_m$  are inapproximable on average and not just in the worst case. The following Lemma gives a stability result for  $A_r$  :

► **Lemma 3.4.** *Let  $V = \{\mathbf{v}_1, \dots, \mathbf{v}_n\} \subseteq \mathbb{F}_2^n$  be a set of vectors of Hamming weight  $w$ . Assume the new set  $V' = \{\mathbf{v}'_1, \dots, \mathbf{v}'_n\} \subseteq \mathbb{F}_2^n$  is obtained from  $V$  by altering at most  $d$  coordinates of each  $\mathbf{v}_i$ , where  $d < w$ ; then for all  $r$ ,*

$$A_r(V') \leq \left( \sqrt{A_r(V)} + \sqrt{\frac{d}{w}} \right)^2. \quad (13)$$

**Proof.** Let  $\mathbf{v}, \mathbf{v}'$  be arbitrary binary vectors such that the Hamming weight of  $\mathbf{v}$  is  $w$  and the Hamming distance between  $\mathbf{v}$  and  $\mathbf{v}'$  is at most  $d$ . It is not hard to see that after the embedding in the real space we have

$$(\mathbf{v}, \mathbf{v}') \geq \sqrt{1 - \frac{d}{w}}. \tag{14}$$

The minimum in (14) is attained by a vector  $\mathbf{v}'$  of Hamming weight  $w - d$ , where the support of  $\mathbf{v}'$  is a subset of the support of  $\mathbf{v}$ . Let  $W$  be an  $r$ -dimensional real space in  $\mathbb{R}^n$  that attains the maximum in (6) for approximating the set  $V'$ . Let  $\mathbf{w}_1, \dots, \mathbf{w}_n$  be a family of unit vectors in  $W$  such that for all  $i \in [n]$ ,  $(\mathbf{v}'_i, \mathbf{w}_i)^2 \geq A_r(V')$ . Let  $A = \sqrt{A_r(V)}$ . By definition of  $A$  there exists  $i \in [n]$  such that

$$(\mathbf{v}_i, \mathbf{w}_i) \leq A. \tag{15}$$

We introduce notation for angles between vectors  $\mathbf{v}_i, \mathbf{v}'_i$ , and  $\mathbf{w}_i$ . Let

$$\alpha = \angle(\mathbf{v}'_i, \mathbf{w}_i), \quad \beta = \angle(\mathbf{v}_i, \mathbf{v}'_i), \quad \gamma = \angle(\mathbf{v}_i, \mathbf{w}_i).$$

Clearly  $\alpha, \beta \in [0, \pi/2]$  and  $\alpha \geq \gamma - \beta$ . First suppose that  $\gamma - \beta \geq 0$ ; then

$$(\mathbf{v}'_i, \mathbf{w}_i) = \cos \alpha \leq \cos \gamma \cos \beta + \sin \gamma \sin \beta \leq \max\{0, \cos \gamma\} + \sin \beta \leq A + \sqrt{\frac{d}{w}},$$

where the last inequality follows from (14) and (15). Now note that if  $0 \leq \gamma \leq \beta \leq \pi/2$ ; then

$$(\mathbf{v}'_i, \mathbf{w}_i) \leq 1 \leq \cos \gamma + \sin \beta \leq A + \sqrt{\frac{d}{w}}.$$

The inequality (13) follows. ◀

The above Lemma and identity (12) yield

► **Proposition 3.5.** *Let  $V_m$  be an  $n \times n$  matrix of a combinatorial design with the parameters of a geometric design  $G_m$ . Assume  $m = \frac{1}{\epsilon}$  is fixed and  $n$  grows to infinity. Let  $V'_m$  be obtained from  $V_m$  by altering each row in up to  $O(n^{1-2\epsilon})$  coordinates. Let  $r = \omega(n^{1-\epsilon})$ . We have*

$$A_r(V'_{1/\epsilon}) \approx \frac{r}{n}. \tag{16}$$

### 3.3 The conjecture and rigidity implications

We now introduce our main conjecture stating that low-dimensional  $\mathbb{F}_2$ -linear spaces  $L$  always have a somewhat large value of  $A_r$  and show how this conjecture implies rigidity of design matrices  $V_m$ . We begin with a formal definition of rigidity.

► **Definition 3.6.** Let  $V$  be an  $n \times n$  matrix over a field  $\mathbb{F}$ . We say that  $V$  is  $(r, d)$ -rigid; if for every matrix  $V'$  that differs from  $V$  in at most  $d$  coordinates in each row, we have  $\text{rank}_{\mathbb{F}} V \geq r$ .

► **Conjecture 3.7.** *There exists positive constants  $\alpha, \delta$ , and  $\epsilon = \frac{1}{m}$  (for an integer  $m$ ) such that for all linear spaces  $L \subseteq \mathbb{F}_2^n$  where  $\dim L \leq n^{2\epsilon+\delta}$ , for some  $r = \omega(n^{1-\epsilon})$ ,*

$$A_r(L) \geq (1 + \alpha) \frac{r}{n}. \tag{17}$$

The conjecture above trivially implies  $(n^{2\epsilon+\delta}, n^{1-2\epsilon})$ -rigidity of design matrices  $V_m$  over the field  $\mathbb{F}_2$ . If some matrix  $V'_m$  had rank below  $n^{2\epsilon+\delta}$  after  $O(n^{1-2\epsilon})$  alterations in each row; then its rows would belong to a  $n^{2\epsilon+\delta}$ -dimensional linear space over  $\mathbb{F}_2$ , and thus have non-trivial approximation measure, contradicting Proposition 3.5. Currently we can only prove the conjecture for all linear spaces  $L$  with  $\dim L = o(n^\epsilon \log n)$ . (Theorem 5.5).

► **Remark.** Replacing the condition  $\dim(L) \leq n^{2\epsilon+\delta}$  in the Conjecture above by the condition  $\dim(L) \leq O\left(n^{\epsilon \log(\frac{1}{\epsilon}+1)}\right)$  makes the Conjecture invalid as by formula (3) matrices  $V_{1/\epsilon}$  may have  $\mathbb{F}_2$  rank of  $O\left(n^{\epsilon \log(\frac{1}{\epsilon}+1)}\right)$ .

#### 4 Low dimensional approximations from bounded triangular rank

In this and the following two Sections we prove our main results regarding approximability of low-dimensional  $\mathbb{F}_2$ -linear spaces  $L$ . In the current Section we deal with low-dimensional approximations and obtain a bound for  $A_1(L)$ . All results obtained in this Section apply not just to  $\mathbb{F}_2$ -linear spaces but to all families of vectors that have bounded triangular rank.

► **Definition 4.1.** Let  $T = \{\mathbf{v}_1, \dots, \mathbf{v}_t\}$  be a sequence of binary vectors of dimension  $n$ . We say that  $T$  is a tower of height  $t$  if for all  $j \leq t$ :

$$\text{supp}(\mathbf{v}_j) \not\subseteq \bigcup_{s \leq j-1} \text{supp}(\mathbf{v}_s).$$

Further, let  $V$  be an arbitrary collection of binary vectors. We define the triangular rank of  $V$ , denoted  $\text{trk}(V)$  to be the largest height of a tower that can be constructed from elements of  $V$ .

It is easy to see that for any subset  $V$  of an  $\mathbb{F}_2$ -linear space  $L$  we always have  $\text{trk}(V) \leq \dim L$ . Let  $L \subseteq \mathbb{F}_2^n$ ,  $\text{trk}(L) \leq k$ . For  $i \in [n]$ , let

$$w_i = \min_{\mathbf{v} \in L : i \in \text{supp}(\mathbf{v})} |\text{supp}(\mathbf{v})|. \quad (18)$$

If  $i \in [n]$  does not belong to the support of any vector in  $L$ ; we define  $w_i = \infty$ . We set

$$\mu = \mu(L) = \sum_{i=1}^n w_i^{-1}. \quad (19)$$

Our proof of the following Lemma resembles some of the arguments in [14, Section 2].

► **Lemma 4.2.** *Let  $L \subseteq \mathbb{F}_2^n$ ,  $\text{trk}(L) \leq k$ ; then*

$$\mu(L) \leq k.$$

**Proof.** Assume  $\mu > k$ . We derive a contradiction by exhibiting a collection  $V = \{\mathbf{v}_1, \dots, \mathbf{v}_{k+1}\} \subseteq L$  such that for every  $j \in [k+1]$ , there exists  $i_j \in \text{supp}(\mathbf{v}_j)$ , such that  $i_j \notin \text{supp}(\mathbf{v}_s)$ , for all  $s < j$ . Consider an  $n$ -node hypergraph, where the hyperedges are the supports of the elements of  $L$ . Color all nodes white. Set  $\Phi = \mu$ ,  $V = \emptyset$ . On the  $j$ -th step:

1. We choose a white node  $i$  whose  $w_i$  is the smallest among the white nodes;
2. We set  $\mathbf{v}_j$  to be a weight- $w_i$  element of  $L$  such that  $i \in \text{supp}(\mathbf{v}_j)$ ;
3. We set

$$\Delta = \sum_{s \in \text{supp}(\mathbf{v}_j) \mid s \text{ is white}} w_s^{-1}.$$

It is important to note that  $\Delta$  is necessarily at most 1.

4. We reduce  $\Phi$  by  $\Delta$  and color all nodes  $s \in \text{supp}(\mathbf{v}_j)$  black.

On each step we reduce  $\Phi$  by at most one and increase  $\text{trk}(V)$  by one. Thus after  $k + 1$  steps we necessarily have  $\text{trk}(V) > k$ . ◀

► **Theorem 4.3.** *Let  $L \subseteq \mathbb{F}_2^n$ ,  $\text{trk}(L) \leq k$ ; then*

$$A_1(L) \geq \frac{1}{k}.$$

**Proof.** Let  $\{w_i\}$  and  $\mu$  be as defined in (18) and (19). Fix a vector  $\mathbf{w} \in \mathbb{R}^n$ , where for all  $i \in [n]$ ,  $\mathbf{w}_i = \sqrt{\frac{1}{\mu w_i}}$ . Clearly  $\|\mathbf{w}\| = 1$ . Let  $\mathbf{v} \in L$  be arbitrary,  $|\text{supp}(\mathbf{v})| = w$ . Note that for all  $i \in \text{supp}(\mathbf{v})$ ,  $w_i \leq w$ . It remains to note that

$$(\mathbf{w}, \mathbf{v}) \geq \frac{w\sqrt{1/\mu w}}{\sqrt{w}} = \frac{1}{\sqrt{\mu}} \geq \frac{1}{\sqrt{k}},$$

where the last inequality follows from Lemma 4.2. ◀

Theorem 4.3 exhibits a vast gap between  $A_1(\mathbb{F}_2^n) \approx \frac{1}{n}$  and  $A_1(L) \geq \frac{1}{k}$  for  $\mathbb{F}_2$ -linear spaces  $L$  that have polynomially low dimension  $k$ . This Theorem alone already implies that our main Conjecture 3.7 holds for all linear spaces of dimension up to  $o(n^\epsilon)$ . In fact it shows that even one-dimensional approximations of discrete linear spaces suffice to get this result.

## 5 High dimensional approximations from bounded triangular rank

In this Section we deal with high dimensional approximations and state the implications of our results for the Hamada’s conjecture. Our main result is given by Theorem 5.5. As in the previous Section our arguments apply not just to  $\mathbb{F}_2$ -linear spaces but to all families of vectors that have bounded triangular rank. To simplify notation in this Section we do not distinguish between binary vectors  $\mathbf{v}$  and their support sets  $\text{supp}(\mathbf{v})$ .

► **Definition 5.1.** Let  $L$  be a family of subsets of some universe. Let  $S$  be a subset of the same universe. We say that  $S$  is a  $(c, k)$ -attractor for  $L$  if  $\text{trk}(L) \leq k$  and for all  $\mathbf{v} \in L$  such that  $\mathbf{v} \cap S \neq \emptyset$ ,

$$|\mathbf{v} \cap S| \geq c \cdot \frac{|S|}{k}. \tag{20}$$

Informally, an attractor is a subset of coordinates such that every low weight vector (i.e., a vector of weight around  $\frac{n}{k}$  or less) in  $L$  whose support intersects the subset, intersects it by more than one would expect. Below is the key Lemma of this Section.

► **Lemma 5.2.** *Let  $L$  be a family of binary vectors. Let  $[N]$  be the union of supports of vectors in  $L$ . Let  $\text{trk}(L) \leq k$ . Assume that Hamming weights of all vectors in  $L$  lie in the segment  $[w, 2w]$ . Further assume  $k \geq 2^{5c+2}$  where  $c$  is an integer. Then there exists a  $(c, k)$ -attractor for  $L$  of size at least  $\frac{N}{2^{4c}}$ .*

**Proof.** Note that if  $\frac{N}{2^{4c+2}} < w$ , then the set  $[N]$  is a  $(c, k)$ -attractor for  $L$ . In fact, let  $\mathbf{v} \in L$  be arbitrary. We have

$$|\text{supp}(\mathbf{v}) \cap [N]| \geq w \geq \frac{N}{2^{4c+2}} \geq c \cdot \frac{N}{2^{5c+2}} \geq c \cdot \frac{|[N]|}{k}.$$

Thus without loss of generality we assume

$$\frac{N}{2^{4c+2}} \geq w. \tag{21}$$

We now execute the following simple greedy algorithm that constructs a tower in the family  $L$ .

1. Set the tower  $T$  to be an empty family of sets. Set  $R = [N]$ .
2. WHILE  $R \neq \emptyset$  DO
3.     BEGIN
4.         Identify a set  $\mathbf{v} \in L$  that minimizes  $|\mathbf{v} \cap R|$ ;
5.         Add  $\mathbf{v}$  to the tower  $T$ ;
6.         Drop the elements in  $\mathbf{v}$  from  $R$ ;
7.     END

The algorithm above terminates producing a tower of height at most  $k$ . On step  $j$  the algorithm adds a new vector to  $T$  and reduces the set  $R$  by  $\Delta_j = T \cap R$ . We partition the steps of the algorithm into *stages*. A step falls into stage number  $i$  if in the beginning on the step

$$\frac{N}{2^i} < |R| \leq \frac{N}{2^{i-1}}. \quad (22)$$

Observe that among the first  $4c$  stages there is at least one stage  $i$ , such that the height of  $T$  increases by  $t \leq \frac{k}{4c}$  during that stage. Let  $S$  be the change in the set  $R$  on the stage  $i$ . We have

$$|S| \geq \left( \frac{N}{2^{i-1}} - 2w \right) - \frac{N}{2^i} \geq \frac{N}{2^{i+1}}.$$

The first inequality above follows from the fact that in the beginning of stage  $i$  the size of  $R$  is at least  $\frac{N}{2^{i-1}}$  minus the size of the last step of stage  $(i-1)$  and the fact that every step size is bounded by  $2w$ . The second inequality follows from (21). Let  $a$  be the index of the first step of stage  $i$ . We have

$$\sum_{j=a}^{a+t-1} |\Delta_j| = |S| \geq \frac{N}{2^{i+1}}.$$

Thus at stage  $i$  there exists a step  $j$  such that

$$|\Delta_j| \geq \frac{N}{2^{i+1}} \cdot \frac{4c}{k}.$$

Let  $A$  be the set  $R$  at the beginning of step  $j$ . As step  $j$  belongs to stage  $i$ , by (22) we have

$$|A| \leq \frac{N}{2^{i-1}}.$$

Combining the last two inequalities we get

$$\frac{|A|}{|\Delta_j|} \leq \frac{k}{c}.$$

Therefore by the greedy property of our algorithm for every set  $\mathbf{v} \in L$  that intersects  $A$  we have

$$\frac{|A|}{|A \cap \mathbf{v}|} \leq \frac{k}{c},$$

or equivalently

$$|A \cap \mathbf{v}| \geq c \cdot \frac{|A|}{k}.$$

Thus  $A$  is a  $(c, k)$ -attractor for  $L$  of size at least  $\frac{N}{2^{4c}}$ . ◀

► **Lemma 5.3.** *Let  $L$  be a family of binary vectors,  $\text{trk}(L) \leq k$ . Assume that Hamming weights of all vectors in  $L$  lie in some segment  $[\sqrt{2}w, 2w]$ . Further assume  $k \geq 2^{5c+2}$  where  $c$  is an integer. We have*

$$A_{c \cdot 2^{4c}}(L) \geq \Omega\left(\frac{c}{k}\right), \tag{23}$$

where the constant in  $\Omega$ -notation is absolute.

**Proof.** Set  $[N]$  be the union of supports of vectors in  $L$ . Clearly,  $N \leq 2wk$ . We now construct a basis for the approximating real space.

1.  $\pi = \{\pi_i\}_{i \geq 0}$  is a partition of  $[N]$ . Initially  $\pi$  consists of a single set  $\pi_0 = [N]$  and  $i = 0$ .
2. WHILE  $((i < c \cdot 2^{4c}) \text{ AND } (\pi_0 \neq \emptyset))$  DO
3. BEGIN
4. Identify a  $(c, k)$ -attractor  $\pi_i$  for  $L$  of relative size at least  $\frac{1}{2^{4c}}$ ;
5. Remove elements of  $\pi_i$  from  $\pi_0$  and from all sets  $\mathbf{v} \in L$ ;
6. Add the set  $\pi_i$  to  $\pi$ ;
7. Drop every element  $\mathbf{v}$  such that  $|\mathbf{v}| < w$  from  $L$ ;
8. Increment  $i$ ;
9. END

Lemma 5.2 ensures that the attractor constructed on step 4 above always exists. Observe that by the end of the execution of the algorithm we have

$$|\pi_0| \leq N \cdot \left(1 - \frac{1}{2^{4c}}\right)^{c \cdot 2^{4c}} \leq \frac{N}{c}. \tag{24}$$

Recall that  $\pi = \{\pi_i\}_{i \geq 0}$  is a partition of  $[N]$ . Let  $W$  be the real linear space spanned by binary vectors  $\mathbf{p}_0, \dots, \mathbf{p}_r$  whose supports are elements of this partition. Clearly,  $\dim W \leq c \cdot 2^{4c}$ . We claim that  $W$  approximates all  $\mathbf{v} \in L$  well. Consider two cases:

- $|\mathbf{v} \cap \pi_0| < w$ . At least  $(\sqrt{2} - 1)w$  elements of  $\text{supp}(\mathbf{v})$  fall onto  $(c, k)$ -attractors in  $\pi$ . To approximate  $\mathbf{v} \in L$  consider the set  $J = \{j \mid \mathbf{v} \cap \pi_j \neq \emptyset \text{ and } j \neq 0\}$ . For all  $j \in J$  by (20) we have

$$|\mathbf{v} \cap \pi_j| \geq c \cdot \frac{|\pi_j|}{k}.$$

Therefore

$$|\pi_j| \leq |\mathbf{v} \cap \pi_j| \cdot \frac{k}{c}. \tag{25}$$

Consider the vector  $\mathbf{p} = \sum_{j \in J} \mathbf{p}_j$ . Summing (25) over all  $j \in J$  we obtain

$$\text{wt}(\mathbf{p}) \leq 2w \cdot \frac{k}{c},$$

where  $\text{wt}(\mathbf{p})$  denotes the Hamming weight. Thus

$$\left(\frac{\mathbf{p}}{\|\mathbf{p}\|}, \frac{\mathbf{v}}{\|\mathbf{v}\|}\right)^2 \geq \frac{|\mathbf{v} \cap \text{supp}(\mathbf{p})|^2}{\text{wt}(\mathbf{v})} \cdot \frac{c}{2wk} \geq \frac{(\sqrt{2} - 1)^2}{4} \cdot \frac{c}{k}.$$

- $|\mathbf{v} \cap \pi_0| \geq w$ . By (24) we have  $|\pi_0| \leq \frac{2wk}{c}$ . Consider the binary real vector  $\mathbf{p}$  whose support is  $\pi_0$ . We have

$$\left(\frac{\mathbf{p}}{\|\mathbf{p}\|}, \frac{\mathbf{v}}{\|\mathbf{v}\|}\right)^2 \geq \frac{w^2}{2w} \cdot \frac{c}{2wk} = \frac{c}{4k}. \quad \blacktriangleleft$$

In what follows all log's are base 2 unless otherwise specified.

► **Theorem 5.4.** *Let  $L \subseteq \mathbb{F}_2^n$ ,  $\text{trk}(L) \leq k$ . Assume  $k \geq 2^{5c+2}$  where  $c$  is an integer. We have*

$$A_{2^c \cdot 2^{4c \cdot \log n}}(L) \geq \Omega\left(\frac{c}{k}\right), \tag{26}$$

where the constant in  $\Omega$ -notation is absolute.

**Proof.** Partition the set  $L$  into  $2 \log n$  subsets  $L_1, \dots, L_{2 \log n}$  where every set  $L_i$  contains elements of  $L$  whose Hamming weight is between  $2^{(i-1)/2}$  and  $2^{i/2}$ . Apply Lemma 5.3 to each  $L_i$ . Consider the joint span of  $2 \log n$  resulting real spaces to approximate  $L$ . ◀

► **Theorem 5.5.** *Let  $L \subseteq \mathbb{F}_2^n$ ,  $\text{trk}(L) \leq k$ ; then*

■ *For all  $\tau > 0$  and sufficiently large  $k$  and  $n$ ,*

$$A_{n^\tau}(L) \geq \Omega\left(\frac{\log k}{k}\right), \tag{27}$$

where the constant in the  $\Omega$ -notation depends only on  $\tau$ .

■ *The bullet above implies that for all  $\alpha$  and all  $\epsilon > 0$  our main Conjecture 3.7 holds for all linear spaces  $L$ , where  $\dim L = o(n^\epsilon \log n)$ .*

**Proof.** We start with the first bullet. Set  $c = \left\lfloor \min \left\{ \frac{\tau}{8} \log k, \frac{\log k - 2}{5} \right\} \right\rfloor$ . Theorem 5.4 yields

$$A_{\frac{\tau}{4} \log k \log n k^{\tau/2}}(L) \geq \Omega\left(\frac{\log k}{k}\right),$$

which immediately yields (27) for large enough  $n$ .

We proceed to the second bullet. Let  $k = \dim L = \beta(n)n^\epsilon \log n$ , where  $\beta(n) \rightarrow 0$  but  $\beta \log n$  grows. Fix an arbitrary  $\tau < 1 - \epsilon$ . By (27)

$$A_{n^\tau}(L) \geq \frac{c \log k}{k},$$

for some constant  $c$ . Set  $r(n) = \frac{c\epsilon n^{1-\epsilon}}{\beta(n)(1+\alpha)}$ . Observe that for sufficiently large  $n$ ,

$$A_r(L) \geq A_{n^\tau}(L) \geq \frac{c \log k}{k} \geq \frac{c\epsilon \log n}{\beta n^\epsilon \log n} = \frac{c\epsilon}{\beta n^\epsilon} = (1 + \alpha) \frac{r}{n}.$$

This concludes the proof. ◀

The following Corollary gives the implication of Theorem 5.5 for the triangular rank of combinatorial designs.

► **Corollary 5.6.** *Let  $m = \frac{1}{\epsilon}$  and let  $V_m$  be the  $n \times n$  incidence matrix of a combinatorial design that has the parameters of the geometric design  $G_{m,q}$ . We have*

$$\text{trk}(V_m) = \Omega(n^\epsilon \log_2 n), \tag{28}$$

where  $\text{trk}(V_m)$  denotes the triangular rank of the collection of rows of  $V_m$ .

**Proof.** Let  $\text{trk}(V_m) = k$ . Set  $r = n^{1-\epsilon}$ . From Corollary 3.3 we have  $A_r(V_m) \lesssim \frac{2}{n^\epsilon}$ . However from Theorem 5.5 we have  $A_r(V_m) \geq \Omega\left(\frac{\log k}{k}\right)$ . Therefore

$$\frac{2}{n^\epsilon} \geq \Omega\left(\frac{\log k}{k}\right).$$

Thus  $k = \Omega(n^\epsilon \log_2 n)$ . ◀

Note that triangular rank of  $V_m$  gives a lower bound for the rank of  $V_m$  over any field.



**6 High dimensional approximations from short dual vectors**

In this Section we establish Conjecture 3.7 for a certain class a linear spaces called cut-spaces and give a simpler proof of a slightly weaker version of Theorem 5.5. In both of these results we use low weight vectors in the dual space of an  $\mathbb{F}_2$ -linear space  $L$  to construct the approximating real space for  $L$ . As in the previous Section, we often write  $\mathbf{v}$  to denote both a vector  $\mathbf{v}$  and its support set  $\text{supp}(\mathbf{v})$ .

► **Definition 6.1.** Let  $L$  be a family of subsets of  $[n]$ . We say that the  $r$ -partition  $\pi = \bigsqcup_{j \leq r} \pi_j$  of  $[n]$  is attractive for  $L$ , if for some constant  $\alpha > 0$  for every  $\mathbf{v} \in L$  we have

$$\left| \bigsqcup_{j : \mathbf{v} \cap \pi_j \neq \emptyset} \pi_j \right| \leq \frac{|\text{supp}(\mathbf{v})|}{(1 + \alpha)} \cdot \frac{n}{r}. \tag{29}$$

► **Lemma 6.2.** Let  $L$  be a set of  $n$ -dimensional binary vectors. Suppose there exists an  $r$ -partition of  $[n]$  that is attractive for  $L$ ; then  $A_r(L) \geq (1 + \alpha) \frac{r}{n}$ .

**Proof.** Let  $W$  be the real linear space spanned by binary vectors  $\mathbf{p}_1, \dots, \mathbf{p}_r$  whose supports are elements of the attractive  $r$ -partition. To approximate  $\mathbf{v} \in L$  consider the vector  $\mathbf{p} = \sum_{j : \mathbf{v} \cap \pi_j \neq \emptyset} \mathbf{p}_j$ . We have

$$\text{wt}(\mathbf{p}) \leq \frac{\text{wt}(\mathbf{v})}{(1 + \alpha)} \cdot \frac{n}{r}.$$

Thus  $(\mathbf{p} / \|\mathbf{p}\|, \mathbf{v})^2 \geq (1 + \alpha) \frac{r}{n}$ . ◀

**6.1 Approximating cut spaces**

A cut space is a subspace of  $\mathbb{F}_2^n$  that has a  $k \times n$  generator matrix where every column has weight two. Equivalently, a cut space is defined by a  $k$ -node graph  $G$  with  $n$  edges. Elements of the cut space are incidence vectors of cuts in the graph. Elements of the dual space are incidence vectors of even degree subgraphs of  $G$ . In what follows we restrict our attention to connected graphs  $G$ . For such graphs the dimension of the corresponding cut space is  $k - 1$ . We now argue that cut spaces satisfy Conjecture 3.7.

► **Theorem 6.3.** Let  $L \subseteq \mathbb{F}_2^n$  be a cut space,  $\dim L \leq o\left(\frac{n}{\log n}\right)$ . Then for some  $r = \Theta\left(\frac{n}{\log n}\right)$  and  $\alpha > 0$  we have

$$A_r(L) \geq (1 + \alpha) \frac{r}{n}.$$

**Proof.** We rely on the fact that any graph with  $k$  nodes and  $n$  edges contains a cycle of length at most  $2 \log n$  provided  $n \geq 3k$ . We consider the graph  $G$  corresponding to  $L$ . We construct a family  $\pi$  of disjoint subsets of edges of  $G$  (coordinates of  $L$ ). We start by executing the following simple algorithm:

1. Start with an empty family of sets  $\pi$ .
2. WHILE  $n \geq 3k$  DO
3.     BEGIN
4.         Identify a cycle  $C$  in  $G$ ,  $|C| \leq 2 \log n$ ;
5.         Include  $C$  into  $\pi$  as a new set;
6.         Drop edges in  $C$  from  $G$ ;
7.     END

Our next goal is to make sure that most sets in  $\pi$  have approximately the same size. Firstly, we repeatedly join together any two sets in  $\pi$ , if the sum of their sizes is below  $2 \log n$ . Secondly, we drop the smallest set from  $\pi$ . Now every set in  $\pi$  has size in the range  $[\log n, 2 \log n]$ . We fix a small  $\delta > 0$  and consider two alternatives:

- *The average size of a set in  $\pi$  is larger than  $(1 + \delta) \log n$ .* We extend  $\pi$  to become a partition of the set  $[n]$  by including all remaining coordinates as singleton sets. Let  $r = |\pi|$ . We have

$$r \leq \frac{n}{(1 + \delta) \log n} + 3k + \log n \lesssim \frac{n}{(1 + \delta) \log n}.$$

We claim that  $\pi$  is attractive for  $L$ . Observe that every element  $\mathbf{v} \in L$  intersects each non-singleton element of  $\pi$  in an even number of coordinates. Therefore we have

$$\left| \bigsqcup_{j : \mathbf{v} \cap \pi_j \neq \emptyset} \pi_j \right| \leq \frac{\text{wt}(\mathbf{v})}{2} \cdot 2 \log n = \text{wt}(\mathbf{v}) \cdot \log n.$$

It remains to note that

$$\text{wt}(\mathbf{v}) \cdot \frac{n}{r} \gtrsim (1 + \delta) \cdot \text{wt}(\mathbf{v}) \cdot \log n.$$

- *The average size of a set in  $\pi$  is below  $(1 + \delta) \log n$ .* The fraction of sets of size above  $1.5 \log n$  is at most  $2\delta$ . We pair up sets of size less than  $1.5 \log n$  arbitrarily (possibly dropping one set). We replace pairs by their unions. This leads us to a new family of sets, where the size of each set is in the range  $[1.5 \log n, 3 \log n]$  and the average size is above  $1.5(1 + \delta') \log n$ . Here we apply the argument from the previous bullet.

This concludes the proof. ◀

## 6.2 Approximating general $\mathbb{F}_2$ -linear spaces

We now apply the method used in the previous Section to approximate cut spaces to generic linear spaces.

- **Theorem 6.4.** *Let  $L \subseteq \mathbb{F}_2^n$  be a linear space,  $\dim L = k$ ,  $k \leq n^{\frac{1}{2}-\beta}$ . Then for some  $r = \Theta\left(\frac{n}{k} \log k\right)$  and  $\alpha > 0$ ,*

$$A_r(L) \geq (1 + \alpha) \frac{r}{n}. \tag{30}$$

**Proof.** Fix  $\epsilon > 0$  such that  $k^{1+\epsilon} = o\left(\frac{n}{k} \log k\right)$ . We rely on the fact that by the Hamming bound for any linear subspace of  $\mathbb{F}_2^m$  of dimension  $k$ , there is a dual vector of weight at most  $ck / \log k$  provided  $k^{1+\epsilon} \leq m$ , for a universal constant  $c$ . We construct a family  $\pi$  of disjoint subsets of  $[n]$ . We start by executing the following simple algorithm:

1. Start with an empty family of sets  $\pi$ .
2. WHILE  $n \geq k^{1+\epsilon}$  DO
3.     BEGIN
4.         Identify a light dual vector  $\mathbf{v}$ ,  $\text{wt}(\mathbf{v}) \leq \frac{ck}{\log k}$ ;
5.         Include  $\text{supp}(\mathbf{v})$  into  $\pi$  as a new set;
6.         Drop the coordinates in  $\text{supp}(\mathbf{v})$  from  $[n]$ . Reduce  $n$  appropriately.
7.     END

Our next goal is to make sure that most sets in  $\pi$  have approximately the same size. Firstly, we repeatedly join together any two sets in  $\pi$ , if the sum of their sizes is below  $2ck/\log k$ . Secondly, we drop the smallest set from  $\pi$ . Now every set in  $\pi$  has size in the range  $[ck/\log k, 2ck/\log k]$ . We fix a small  $\delta > 0$  and consider two alternatives:

- *The average size of a set in  $\pi$  is larger than  $(1 + \delta)ck/\log k$ .* We extend  $\pi$  to become a partition of the set  $[n]$  by including all remaining coordinates as singleton sets. Let  $r = |\pi|$ . We have

$$r \leq \frac{n \log k}{(1 + \delta)ck} + k^{1+\epsilon} + \frac{ck}{\log k} \lesssim \frac{n \log k}{(1 + \delta)ck}.$$

We claim that  $\pi$  is attractive for  $L$ . Observe that every element  $\mathbf{v} \in L$  intersects each non-singleton element of  $\pi$  in an even number of coordinates. Therefore we have

$$\left| \bigsqcup_{j : \mathbf{v} \cap \pi_j \neq \emptyset} \pi_j \right| \leq \frac{\text{wt}(\mathbf{v})}{2} \cdot \frac{2ck}{\log k} = \text{wt}(\mathbf{v}) \cdot \frac{ck}{\log k}.$$

It remains to note that

$$\text{wt}(\mathbf{v}) \cdot \frac{n}{r} \gtrsim \text{wt}(\mathbf{v}) \cdot (1 + \delta) \cdot \frac{ck}{\log k}.$$

- *The average size of a set in  $\pi$  is below  $(1 + \delta)ck/\log k$ .* The fraction of sets of size above  $1.5ck/\log k$  is at most  $2\delta$ . We pair up sets of size less than  $1.5ck/\log k$  arbitrarily (possibly dropping one set). We replace pairs by their unions. This leads us to a new family of sets, where the size of each set is in the range  $[1.5ck/\log k, 3ck/\log k]$  and the average size is above  $1.5(1 + \delta')ck/\log k$ . Here we apply the argument from the previous bullet.

This concludes the proof. ◀

Similarly to Theorem 5.5, Theorem 6.4 above can be used to argue that Conjecture 3.7 holds for all linear spaces of dimension up to  $o(n^\epsilon \log n)$ , i.e., if we set  $k = \beta(n) \cdot n^\epsilon \log n$ , where  $\beta(n) \rightarrow 0$ , and  $r = \Theta\left(\frac{n}{k} \log n\right) = \Theta\left(\frac{n^{1-\epsilon}}{\beta}\right)$ ; then (30) yields

$$A_r(L) \geq (1 + \alpha) \frac{r}{n}. \tag{31}$$

Theorem 5.5 however presents a stronger result. Firstly, in the proof of Theorem 5.5 we use real spaces of dimension as low as  $n^\tau$  to arrive at the bound (31) for some  $r = \omega(n^{1-\epsilon})$ . This leaves plenty of room for potential further improvements. Secondly, Theorem 5.5 applies to all sets of vectors of bounded triangular rank, while Theorem 6.4 only deals with  $\mathbb{F}_2$ -linear spaces.

## 7 Relation to natural proofs

In [16] Razborov and Rudich introduced the natural proofs barrier for proving lower bounds for computational complexity of Boolean functions. Stated informally their results say that if a certain property of Boolean functions is shown to imply hardness then; either typical (random) functions do not have this property, or the property should be hard to recognize, or some well accepted hardness conjectures are invalid. While the theory developed in [16] deals with properties of Boolean functions one can make an analogy in the linear setting by defining a natural property of a matrix to be a property that holds for most matrices and can be verified efficiently [1]. In this case however the respective "hardness conjectures" are not as standard.

In light of the above it is interesting to ask if establishing our main Conjecture 3.7 for some  $\epsilon$  would necessarily certify rigidity of random binary matrices where each element is set to 1 independently with probability  $n^{-\epsilon}$ . The answer to this seems to depend on the value of  $\alpha$  for that one proves the Conjecture. If  $\alpha$  is sufficiently small; then rigidity of random matrices would likely not be implied. The following Theorem shows that unlike the rows of a combinatorial design  $V_{1/\epsilon}$ , the rows of a random binary matrix of density  $n^{-\epsilon}$  with high probability admit a non-trivial approximation on average even in the regime of a fairly large dimension of the approximating real space.

► **Theorem 7.1.** *Let  $V$  be a random  $n \times n$  matrix of zeros and ones where every entry is set to 1 independently with probability  $p = n^{-\epsilon}$ ,  $\epsilon < \frac{1}{4}$ . Let  $B$  be the real matrix, where the rows of  $B$  are the normalized elements of  $V$ . Let  $\lambda_1 \geq \dots \geq \lambda_n \geq 0$  be the eigenvalues of  $BB^t$ . There exists  $\alpha > 0$  such that with high probability over the choice of  $V$  for all  $r = o(n)$  we have,*

$$\max_{\dim W \leq r} \mathbb{E}_{\mathbf{v} \in V} \|\text{Pr}_W(\mathbf{v})\|^2 = \frac{1}{n} \sum_{i \leq r} \lambda_i \geq (1 + \alpha) \frac{r}{n}. \tag{32}$$

**Proof.** The equality above is given by (10). We need to prove the inequality, which is a straightforward corollary of the Marchenko-Pastur law [4] determining the limiting behavior of the spectral distribution of large inner product matrices. We state a special case of this law that will suffice for our purposes (see Theorem 3.10 in [4]).

Let  $\{M_n\}$  be a sequence of  $n \times n$  random matrices, such that the entries of  $M_n$  are i.i.d. random variables with expectation  $\mu_n$  and variance 1. Let  $\lambda_1, \dots, \lambda_n$  be the eigenvalues of  $\frac{1}{n} M_n M_n^t$ . Then, for any  $0 \leq a \leq 4$  holds, with probability 1, that

$$\frac{\#\{i : \lambda_i \geq a\}}{n} \longrightarrow \frac{1}{2\pi} \int_a^4 \sqrt{\frac{4-x}{x}} dx \tag{33}$$

Fix  $a = 2$ . Let  $c = \frac{1}{2\pi} \int_2^4 \sqrt{\frac{4-x}{x}} dx = \frac{\pi-2}{2\pi} \approx 0.18$ . Taking  $M_n = \frac{1}{\sqrt{p(1-p)}} \cdot V$ , we observe that, with probability tending to one with  $n$ , at least  $cn - o(n)$  eigenvalues of  $\frac{1}{p(1-p)n} VV^t$  are greater or equal 2.

To complete the proof, we will argue that the spectral distribution of  $\frac{1}{p(1-p)n} VV^t$  is close to that of  $BB^t$ . In fact, a special case of the perturbation inequality A.41 in [4] states that for any two symmetric  $n \times n$  matrices  $X$  and  $Y$ , with eigenvalues  $\lambda_1, \dots, \lambda_n$  and  $\mu_1, \dots, \mu_n$ , and for any real number  $a$  holds that

$$\left| \frac{\#\{i : \lambda_i \geq a\}}{n} - \frac{\#\{i : \mu_i \geq a\}}{n} \right| \leq \left( \frac{1}{n} \cdot \text{Tr} \left( (X - Y)^2 \right) \right)^{1/3} \tag{34}$$

We will apply this inequality to slightly perturbed versions of the matrices  $\frac{1}{p(1-p)n} VV^t$  and  $BB^t$ . The goal of this modification would be to cancel the problematic effect of the maximal eigenvalues of the two matrices. With this in mind, we set  $U = V - pJ$ , where  $J$  is the all-1 matrix. Let  $W = (w_{ij}) = UU^t$ . We take  $X = \frac{1}{p(1-p)n} W$ .

Next, we consider the norms of the rows of  $V$ . Since  $V$  is a 0-1 matrix, the norm of its  $i^{\text{th}}$  row is  $\sqrt{r_i}$ , where  $r_i$  is the row sum. We take  $Y = \left( \frac{w_{ij}}{\sqrt{r_i r_j}} \right)$ .

Note that the matrices  $X$  and  $Y$  are rank-1 perturbations of  $\frac{1}{p(1-p)n} VV^t$  and  $BB^t$ . It is well-known that if two matrices differ by a matrix of rank 1, their eigenvalues interlace. Hence, using the perturbed matrices changes the LHS of (34) by at most an additive factor of  $O(1/n)$ , which we may ignore.

In the following analysis we may and will assume all  $r_i$  to lie in the interval  $np \pm t\sqrt{np \log n}$ , for a sufficiently large absolute constant  $t$ , since this holds with probability tending to one with  $n$ , by the Chernoff bound.

With this assumption, we can bound the distance between the entries of  $X$  and  $Y$  as follows.

$$(x_{ij} - y_{ij})^2 = \left( \frac{w_{ij}}{p(1-p)n} - \frac{w_{ij}}{\sqrt{r_i r_j}} \right)^2 \leq O\left(\frac{1}{n^2}\right) \cdot w_{ij}^2$$

In the last inequality we have used the fact that  $np^2 \gg \sqrt{n \log n}$ , which we may do since  $p = n^{-\epsilon}$  and, by assumption,  $\epsilon < 1/4$ .

To complete the argument about proximity of the spectral distributions of  $X$  and  $Y$ , we need to estimate from above the  $\ell_2$  norm of  $W$ . Note first that by (33) we may assume all the eigenvalues of  $X$  to lie between 0 and 4. Hence  $\sum_{i,j=1}^n x_{ij}^2 = O(n)$ . Since  $W = p(1-p)n \cdot X$ , we deduce  $\sum_{i,j=1}^n w_{ij}^2 \leq n^2 p^2 \cdot \sum_{i,j=1}^n x_{ij}^2 \leq O(n^3 p^2)$ .

Finally, we can estimate the RHS of (34) from above as follows:

$$\left( \frac{1}{n} \cdot \text{Tr}((X - Y)^2) \right)^{1/3} \leq O\left(\frac{1}{n}\right) \cdot \left( \sum_{i,j=1}^n w_{ij}^2 \right)^{1/3} \leq O(p^{2/3}) = o(n)$$

We deduce that at least  $cn - o(n)$  eigenvalues of the matrices  $Y$  and (hence)  $BB^t$  are greater or equal 2, for an absolute constant  $c \approx 0.18$ . The claim of the theorem follows. ◀

## 8 Conclusions

In this paper we suggested a new path to establishing rigidity of design matrices over the field  $\mathbb{F}_2$ . Our approach is centered around the conjecture that says that after the natural "normalizing" embedding of the Boolean cube into  $\mathbb{R}^n$ , low dimensional  $\mathbb{F}_2$ -linear spaces exhibit some tiny amount of resemblance to real linear spaces. In particular it is easier to approximate them by Euclidian linear spaces than to approximate all of the Boolean cube. We showed that the conjecture is indeed true (by a huge margin) when approximating real spaces are of low dimension. However our approximability results for high-dimensional real spaces are not strong enough.

Currently it feels that the weakness of our results stems from the fact in that we use relatively little combinatorial structure of  $\mathbb{F}_2$ -linearity. In particular our strongest result (Theorem 5.5) applies to all sets of bounded triangular rank. Note that while it is plausible that one can make further progress based just on triangular rank; one cannot establish Conjecture 3.7 in such generality.

► **Remark.** Replacing the condition  $\dim(L) \leq n^{2\epsilon+\delta}$  in the Conjecture 3.7 by the condition  $\text{trk}(L) \leq n^{2\epsilon+\delta}$  makes the Conjecture invalid.

**Proof.** Let  $m = \frac{1}{\epsilon}$  be an integer. Let  $V'_m$  be a matrix that is obtained from  $V_m$  by independently flipping every zero entry to one with probability  $n^{-2\epsilon}$ . It is not hard to see that with overwhelming probability  $V'_m$  does not contain an all-zeros minor of size  $\Omega(n^{2\epsilon} \log n)$ . Thus  $\text{trk}(V'_m) = O(n^{2\epsilon} \log n)$ . However Proposition 3.5 implies that for any  $r = \omega(n^{1-\epsilon})$ , we have  $A_r(V'_m) \approx \frac{r}{n}$ . Thus rows of  $V'_m$  give a counterexample to this stronger version of the Conjecture. ◀

**Acknowledgement.** We would like to thank Noga Alon, Swastik Kopparty, and Mark Rudelson for many helpful discussions regarding this work.

## References

- 1 Michael Alekhovich. More on average case vs. approximation complexity. In *44th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 298–307, 2003.
- 2 Noga Alon and Gil Cohen. On rigid matrices and U-polynomials. In *28th IEEE Computational Complexity Conference (CCC)*, pages 197–206, 2013.
- 3 Noga Alon, Rina Panigrahy, and Sergey Yekhanin. Deterministic approximation algorithms for the nearest codeword problem. In *13th International Workshop on Randomization and Computation (RANDOM)*, volume 5687 of Lecture Notes in Computer Science, pages 339–351, 2009.
- 4 Zhidong Bai and Jack Silverstein. *Spectral Analysis of Large Dimensional Random Matrices*. Springer, New York, 2010.
- 5 Zeev Dvir. On matrix rigidity and locally self-correctable codes. *Computational Complexity*, 20:367–388, 2011.
- 6 Joel Friedman. A note on matrix rigidity. *Combinatorica*, 13:235–239, 1993.
- 7 Noboru Hamada. On the  $p$ -rank of the incidence matrix of a balanced or partially balanced incomplete block design and its applications to error-correcting codes. *Hiroshima Mathematical Journal*, 3:154–226, 1973.
- 8 R. S. Ismagilov.  $n$ -dimensional width of compact in Hilbert space. *Journal of Functional Analysis and Applications*, 2:32–39, 1968.
- 9 Stasys Jukna. *Extremal combinatorics*. Springer, Berlin, Heidelberg, New York, 2001.
- 10 Dieter Jungnickel and Vladimir Tonchev. Polarities, quasi-symmetric designs, and hamada conjecture. *Designs, Codes, and Cryptography*, 51:131–140, 2009.
- 11 Boris Kashin and Alexander Razborov. Improved lower bounds on the rigidity of Hadamard matrices. *Mathematical Notes*, 63:471–475, 1998.
- 12 Satyanarayana Lokam. Spectral methods for matrix rigidity with applications to size-depth trade-offs and communication complexity. *Journal of Computer and System Sciences*, 63:449–473, 2001.
- 13 Satyanarayana Lokam. Complexity lower bounds using linear algebra. *Foundations and Trends in Theoretical Computer Science*, 4:1–155, 2009.
- 14 Ilan Newman and Yuri Rabinovich. On multiplicative  $\lambda$ -approximations and some geometric applications. *SIAM Journal on Computing*, 42:885–883, 2013.
- 15 Alexander Razborov. On rigid matrices, 1989. Manuscript. In Russian.
- 16 Alexander Razborov and Steven Rudich. Natural proofs. *Journal of Computer and System Sciences*, 55:24–35, 1997.
- 17 Shubhangi Saraf and Sergey Yekhanin. Noisy interpolation of sparse polynomials, and applications. In *26th IEEE Computational Complexity Conference (CCC)*, pages 86–92, 2011.
- 18 Rocco Servedio and Emanuele Viola. On a special case of rigidity. In *Electronic Colloquium on Computational Complexity (ECCC)*, TR12-144, 2012.
- 19 Amin Shokrollahi, Daniel Spielman, and Voelker Stemann. A remark on matrix rigidity. *Information Processing Letters*, 64:283–285, 1997.
- 20 Kempton Smith. On the  $p$ -rank of the incidence matrix of points and hyperplanes in a finite projective geometry. *Journal of Combinatorial Theory*, 7:122–129, 1969.
- 21 Vladimir Temlyakov. Nonlinear Kolmogorov widths. *Mathematical Notes*, 63:785–795, 1998.
- 22 Kirill Uskov. *Kolmogorov width of geometric configurations and functional compacts in Hilbert spaces*. PhD thesis, Moscow State Technical University, 2002. in Russian.
- 23 Leslie Valiant. Graph-theoretic arguments in low level complexity. In *6th Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 162–176, 1977.

# On the (Non) NP-Hardness of Computing Circuit Complexity\*

Cody D. Murray and R. Ryan Williams

Computer Science Department, Stanford University  
Stanford, CA, USA  
cdmurray@stanford.edu, rrw@cs.stanford.edu

---

## Abstract

The Minimum Circuit Size Problem (MCSP) is: *given the truth table of a Boolean function  $f$  and a size parameter  $k$ , is the circuit complexity of  $f$  at most  $k$ ?* This is the definitive problem of circuit synthesis, and it has been studied since the 1950s. Unlike many problems of its kind, MCSP is *not* known to be NP-hard, yet an efficient algorithm for this problem also seems very unlikely: for example,  $\text{MCSP} \in \text{P}$  would imply there are no pseudorandom functions.

Although most NP-complete problems are complete under strong “local” reduction notions such as poly-logarithmic time projections, we show that MCSP is *provably not* NP-hard under  $O(n^{1/2-\varepsilon})$ -time projections, for every  $\varepsilon > 0$ . We prove that the NP-hardness of MCSP under (logtime-uniform) AC0 reductions would imply extremely strong lower bounds:  $\text{NP} \not\subseteq \text{P}_{/\text{poly}}$  and  $\text{E} \not\subseteq \text{i.o.-SIZE}(2^{\delta n})$  for some  $\delta > 0$  (hence  $\text{P} = \text{BPP}$  also follows). We show that even the NP-hardness of MCSP under general polynomial-time reductions would separate complexity classes:  $\text{EXP} \neq \text{NP} \cap \text{P}_{/\text{poly}}$ , which implies  $\text{EXP} \neq \text{ZPP}$ . These results help explain why it has been so difficult to prove that MCSP is NP-hard.

We also consider the nondeterministic generalization of MCSP: the Nondeterministic Minimum Circuit Size Problem (NMCSP), where one wishes to compute the *nondeterministic* circuit complexity of a given function. We prove that the  $\Sigma_2\text{P}$ -hardness of NMCSP, even under arbitrary polynomial-time reductions, would imply  $\text{EXP} \not\subseteq \text{P}_{/\text{poly}}$ .

**1998 ACM Subject Classification** F.1.3 Complexity Measures and Classes

**Keywords and phrases** circuit lower bounds, minimum circuit size problem, NP-completeness, projections, reductions

**Digital Object Identifier** 10.4230/LIPIcs.CCC.2015.365

## 1 Introduction

The Minimum Circuit Size Problem (MCSP) is the canonical logic synthesis problem: we are given  $\langle T, k \rangle$  where  $T$  is a string of  $n = 2^\ell$  bits (for some  $\ell$ ),  $k$  is a positive integer (encoded in binary or unary), and the goal is to determine if  $T$  is the truth table of a boolean function with circuit complexity at most  $k$ . (For concreteness, let’s say our circuits are defined over AND, OR, NOT gates of fan-in at most 2.) MCSP is in NP, because any circuit of size at most  $k$  could be guessed nondeterministically in  $O(k \log k) \leq O(n)$  time, then verified on all bits of the truth table  $T$  in  $\text{poly}(2^\ell, k) \leq \text{poly}(n)$  time.<sup>1</sup>

---

\* Supported by NSF CCF-1212372. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

<sup>1</sup> Recall that every Boolean function  $f : \{0, 1\}^\ell \rightarrow \{0, 1\}$  has a circuit of size at most  $k \leq \ell \cdot 2^\ell$ . Hence every instance  $\langle T, k \rangle$  with  $k > 2n \log n$  is automatically a *yes*-instance of MCSP.





MCSP is natural and basic, but unlike thousands of other computational problems studied over the last 40 years, the complexity of MCSP has yet to be determined. The problem could be NP-complete, it could be NP-intermediate, or it could even be in P. (It is reported that Levin delayed publishing his initial results on NP-completeness out of wanting to include a proof that MCSP is NP-complete [3]. More notes on the history of this problem can be found in [15].)

### Lower Bounds for MCSP?

There is substantial evidence that  $\text{MCSP} \notin \text{P}$ . If  $\text{MCSP} \in \text{P}$ , then (essentially by definition) efficient algorithms for MCSP imply that there are no pseudorandom functions. Kabanets and Cai [15] made this critical observation, noting that the hardness of factoring Blum integers implies that MCSP is hard. Allender *et al.* [4] strengthened these results considerably, showing that Discrete Log and many approximate lattice problems from cryptography are solvable in  $\text{BPP}^{\text{MCSP}}$  and Integer Factoring is in  $\text{ZPP}^{\text{MCSP}}$ . (Furthermore, [4] also prove that  $\text{MCSP} \notin \text{AC0}$ .) Allender and Das [5] recently showed that Graph Isomorphism is in  $\text{RP}^{\text{MCSP}}$ , and in fact every problem with statistical zero-knowledge interactive proofs [11] is in promise-BPP with a MCSP oracle.

### NP-Hardness for MCSP?

These reductions indicate strongly that MCSP is not solvable in randomized polynomial time; perhaps it is NP-complete? Evidence for the NP-completeness of MCSP has been less conclusive. The variant of the problem where we are looking for a minimum size DNF (instead of an arbitrary circuit) is known to be NP-complete [10, 6]. Kabanets and Cai [15] show that, if MCSP is NP-complete under so-called “natural” poly-time reductions (where the circuit size parameter  $k$  output by the reduction is a function of only the input length to the reduction) then  $\text{EXP} \not\subseteq \text{P}_{/\text{poly}}$ , and  $\text{E} \not\subseteq \text{SIZE}(2^{\varepsilon n})$  for some  $\varepsilon > 0$  unless  $\text{NP} \subset \text{SUBEXP}$ . Therefore NP-completeness under a restricted reduction type would imply (expected) circuit lower bounds. This doesn’t necessarily show that such reductions do not *exist*, but rather that they will be difficult to construct.

Allender *et al.* [4] show that if  $\text{PH} \subset \text{SIZE}(2^{n^{o(1)}})$  then a (variant of) MCSP is not hard for  $\text{TC}^0$  under AC0 reductions. The generalization  $\text{MCSP}^A$  for circuits with  $A$ -oracle gates has also been studied; it is known for example that  $\text{MCSP}^{\text{QBF}}$  is complete for PSPACE under ZPP reductions [4], and recently Allender, Holden, and Kabanets [7] proved that  $\text{MCSP}^{\text{QBF}}$  is not PSPACE-complete under logspace reductions. They also showed, among similar results, that if there is a set  $A \in \text{PH}$  such that  $\text{MCSP}^A$  is hard for P under AC0 reductions, then  $\text{P} \neq \text{NP}$ .

NP-completeness has been defined for many different reducibility notions: polynomial time, logarithmic space, AC0, even logarithmic time reductions. In this paper, we study the possibility of MCSP being NP-complete for these reducibilities. We prove several new results in this direction, summarized as follows:

1. Under “local” polynomial-time reductions where any given output bit can be computed in  $n^{o(1)}$  time, MCSP is *provably not* NP-complete, contrary to many other natural NP-complete problems. (In fact, even PARITY cannot reduce to MCSP under such reductions: see Theorem 1.2.)
2. Under slightly stronger reductions such as uniform AC0, the NP-completeness of MCSP



would imply  $\text{NP} \not\subseteq \text{P}_{/\text{poly}}$ <sup>2</sup> and  $\text{E} \not\subseteq \text{i.o.-SIZE}(2^{\delta n})$  for some  $\delta > 0$ , therefore  $\text{P} = \text{BPP}$  as well by [14].

3. Under the strongest reducibility notions such as polynomial time, the NP-completeness of MCSP would still imply major separations of complexity classes. For example,  $\text{EXP} \neq \text{ZPP}$  would follow, a major (embarrassingly) open problem.

Together, the above results tell a convincing story about why MCSP has been difficult to prove NP-complete (if that is even true). Part 1 shows that, unlike many textbook reductions for NP-hardness, no simple “gadget-based” reduction can work for proving the NP-hardness of MCSP. Part 2 shows that going only a little beyond the sophistication of textbook reductions would separate P from NP and fully derandomize BPP, which looks supremely difficult (if possible at all). Finally, part 3 shows that even establishing the most relaxed version of the statement “MCSP is NP-complete” requires separating exponential time from randomized polynomial time, a separation that appears quite far from a proof at the present time.

### MCSP is Not Hard Under “Local” Reductions

Many NP-complete problems are still complete under polynomial-time reductions with severe-looking restrictions, such as reductions which only need  $O(\log^c n)$  time to output an arbitrary bit of the output. Let  $t : \mathbb{N} \rightarrow \mathbb{N}$ ; think of  $t(n)$  as  $n^{1-\varepsilon}$  for some  $\varepsilon > 0$ .

► **Definition 1.1.** An algorithm  $R : \Sigma^* \times \Sigma^* \rightarrow \{0, 1, \star\}$  is a  $\text{TIME}(t(n))$  reduction from  $L$  to  $L'$  if there is a constant  $c \geq 1$  such that for all  $x \in \Sigma^*$ ,

- $R(x, i)$  has random access to  $x$  and runs in  $O(t(|x|))$  time for all  $i \in \{0, 1\}^{\lceil 2^c \log_2 |x| \rceil}$ .
- There is an  $\ell_x \leq |x|^c + c$  such that  $R(x, i) \in \{0, 1\}$  for all  $i \leq \ell_x$ , and  $R(x, i) = \star$  for all  $i > \ell_x$ , and
- $x \in L \iff R(x, 1) \cdot R(x, 2) \cdots R(x, \ell_x) \in L'$ .

(Note that  $\star$  denotes an “out of bounds” character to mark the end of the output.) That is, the overall reduction outputs strings of polynomial length, but any desired bit of the output can be printed in  $O(t(n))$  time.  $\text{TIME}(n^{o(1)})$  reductions are powerful enough for almost all NP-completeness results, which have “local” structure transforming small pieces of the input to small pieces of the output.<sup>3</sup> More precisely, an  $O(n^k)$ -time reduction  $R$  from  $L$  to  $L'$  is a *projection* if there is a polynomial-time algorithm  $A$  that, given  $i = 1, \dots, n^k$  in binary,  $A$  outputs either a fixed bit (0 or 1) which is the  $i$ th bit of  $R(x)$  for all  $x$  of length  $n$ , or a  $j = 1, \dots, n$  with  $b \in \{0, 1\}$  such that the  $i$ th bit of  $R(x)$  (for all  $x$  of length  $n$ ) equals  $b \cdot x_j + (1 - b) \cdot (1 - x_j)$ . Skyum and Valiant [17] observed that almost all NP-complete problems are also complete under projections. So for example, we have:

► **Proposition 1** ([17, 16]). *SAT, Vertex Cover, Independent Set, Hamiltonian Path, and 3-Coloring are NP-complete under  $\text{TIME}(\text{poly}(\log n))$  reductions.*

In contrast to the above, we prove that MCSP is *not* complete under  $\text{TIME}(n^{1/3})$  reductions. Indeed there is no local reduction from even the simple language PARITY to MCSP:

<sup>2</sup> After learning of our preliminary results, Allender, Holden, and Kabanets [7] found an alternative proof of the consequence  $\text{NP} \not\subseteq \text{P}_{/\text{poly}}$ .

<sup>3</sup> We say “almost all NP-completeness results” because one potential counterexample is the typical reduction from Subset Sum to Partition: two numbers in the output of this reduction require taking the *sum of all numbers* in the input Subset Sum instance. Hence the straightforward reduction does not seem to be computable even in  $2^{n^{o(1)}}$ -size AC0.

► **Theorem 1.2.** *For every  $\delta < 1/2$ , there is no  $\text{TIME}(n^\delta)$  reduction from  $\text{PARITY}$  to  $\text{MCSP}$ . As a corollary,  $\text{MCSP}$  is not  $\text{AC0}[2]$ -hard under  $\text{TIME}(n^\delta)$  reductions.<sup>4</sup>*

This establishes that  $\text{MCSP}$  cannot be “locally” NP-hard in the way that many canonical NP-complete problems are known to be.

### Hardness Under Stronger Reducibilities

For stronger reducibility notions than sub-polynomial time, we do not yet have unconditional non-hardness results for  $\text{MCSP}$ . (Of course, a proof that  $\text{MCSP}$  is not NP-complete under poly-time reductions would immediately imply  $\text{P} \neq \text{NP}$ .) Nevertheless, we can still prove interesting complexity consequences assuming the NP-hardness of  $\text{MCSP}$  under these sorts of reductions.

► **Theorem 1.3.** *If  $\text{MCSP}$  is NP-hard under polynomial-time reductions, then  $\text{EXP} \neq \text{NP} \cap \text{P}_{/\text{poly}}$ . Consequently,  $\text{EXP} \neq \text{ZPP}$ .*

► **Corollary 1.4.** *If  $\text{MCSP}$  is NP-hard under logspace reductions, then  $\text{PSPACE} \neq \text{ZPP}$ .*

► **Theorem 1.5.** *If  $\text{MCSP}$  is NP-hard under logtime-uniform  $\text{AC0}$  reductions, then  $\text{NP} \not\subseteq \text{P}_{/\text{poly}}$  and  $\text{E} \not\subseteq \text{i.o.-SIZE}(2^{\delta n})$  for some  $\delta > 0$ . As a consequence,  $\text{P} = \text{BPP}$  also follows.*

That is, the *difficulty* of computing circuit complexity would imply *lower bounds*, even in the most general setting (there are *no* restrictions on the polynomial-time reductions here, in contrast with Kabanets and Cai [15]). We conjecture that the consequence of Theorem 1.3 can be strengthened to  $\text{EXP} \not\subseteq \text{P}_{/\text{poly}}$ , and that  $\text{MCSP}$  is (unconditionally) not NP-hard under uniform  $\text{AC0}$  reductions.

### $\Sigma_2$ -Hardness for Nondeterministic $\text{MCSP}$ Implies Circuit Lower Bounds

Intuitively, the difficulty of solving  $\text{MCSP}$  via uniform algorithms should be related to circuit lower bounds against functions defined by uniform algorithms. That is, our intuition is that “ $\text{MCSP}$  is NP-complete” implies circuit lower bounds. We have not yet shown a result like this (but come close with  $\text{EXP} \neq \text{ZPP}$  in Theorem 1.3). However, we can show that  $\Sigma_2\text{P}$ -completeness for the *nondeterministic* version of  $\text{MCSP}$  would imply  $\text{EXP} \not\subseteq \text{P}_{/\text{poly}}$ .

In the Nondeterministic Minimum Circuit Size Problem ( $\text{NMCSP}$ ), we are given  $\langle T, k \rangle$  as in  $\text{MCSP}$ , but now we want to know if  $T$  denotes a boolean function with *nondeterministic* circuit complexity at most  $k$ . It is easy to see that  $\text{NMCSP}$  is in  $\Sigma_2\text{P}$ : nondeterministically guess a circuit  $C$  with a “main” input and “auxiliary” input, nondeterministically evaluate  $C$  on all  $2^\ell$  inputs  $x$  for which  $T(x) = 1$ , then universally verify on all  $2^\ell$  inputs  $y$  satisfying  $T(y) = 0$  that no auxiliary input makes  $C$  output 1 on  $y$ .

We can show that if  $\text{NMCSP}$  is hard even for Merlin-Arthur games, then circuit lower bounds follow.

► **Theorem 1.6.** *If  $\text{NMCSP}$  is MA-hard under polynomial-time reductions, then  $\text{EXP} \not\subseteq \text{P}_{/\text{poly}}$ .*

Vinodchandran [18] studied  $\text{NMCSP}$  for *strong* nondeterministic circuits, showing that a “natural” reduction from SAT or Graph Isomorphism to this problem would have several interesting implications.

<sup>4</sup> Dhiraj Holden and Chris Umans (personal communication) proved independently that there is no  $\text{TIME}(\text{poly}(\log n))$  reduction from SAT to  $\text{MCSP}$  unless  $\text{NEXP} \subset \Sigma_2\text{P}$ .

## 1.1 Intuition

The MCSP problem is a special kind of “meta-algorithmic” problem, where the input describes a function (and a complexity upper bound) and the goal is to essentially compute the circuit complexity of the function. That is, like many of the central problems in theory, MCSP is a problem about computation itself.

In this paper, we apply many tools from the literature to prove our results, but the key idea is to exploit the meta-algorithmic nature of MCSP directly in the assumed reductions to MCSP. We take advantage of the fact that instances of MCSP are written in a rather non-succinct way: the entire truth table of the function is provided. (This observation was also used by Kabanets and Cai [15], but not to the same effect.)

For the simplest example of the approach, let  $L$  be a unary (tally) language, and suppose there is a  $\text{TIME}(\text{poly}(\log n))$  reduction  $R$  from  $L$  to MCSP. The outputs of  $R$  are pairs  $\langle T, k \rangle$ , where  $T$  is a truth table and  $k$  is the size parameter. Because every bit of  $R$  is computable in polylog time, it follows that each truth table  $T$  output by  $R$  can in fact be described by a polylogarithmic size circuit specifying the length of the input instance of  $L$ , and the mechanics of the polylog time reduction used to compute a given bit of  $R$ . Therefore the circuit complexities of *all outputs of  $R$*  are at most polylogarithmic in  $n$  (the input length). Furthermore, the size parameters  $k$  in the outputs of  $R$  on  $n$ -bit inputs are at most  $\text{poly}(\log n)$ , otherwise the MCSP instance is trivially a *yes* instance. That is, the efficient reduction  $R$  itself yields a strong upper bound on the witness sizes of the outputs of  $R$ .

This ability to bound  $k$  from above by a small value based on the existence of an efficient reduction to MCSP is quite powerful. It can also be carried out for more complex languages. For example, consider a polylog time reduction from PARITY to MCSP, where we are mapping  $n$ -bit strings to instances of MCSP. Given any polylog time reduction from PARITY to MCSP, we can construct another polylog time reduction which on every  $n$ -bit string always outputs the *same* circuit size parameter  $k_n$ . That is, we can turn any polylog time reduction into a *natural reduction* in the sense of Kabanets and Cai [15], and apply their work to *general* reductions. (The basic idea is to answer “no” to every bit query of the polylog time reduction, and to then “pad” a given PARITY instance with a few strategically placed zeroes, so that it always satisfies those “no” answers.)

Several of our theorems have the form that, if computing circuit complexity is NP-hard (or nondeterministic circuit complexity is  $\Sigma_2\text{P}$ -hard), then circuit lower bounds follow. This is intriguing to us, as one also expects that *efficient algorithms* for computing circuit complexity also lead to lower bounds! (For example, [15, 13, 19] show that polynomial-time algorithms for MCSP in various forms would imply circuit lower bounds against EXP and/or NEXP.) If a circuit lower bound can be proved to follow from assuming MCSP is NP-intermediate (or NMCSP is  $\Sigma_2\text{P}$ -intermediate), perhaps we can prove circuit lower bounds unconditionally without necessarily resolving the complexity of MCSP.

## 2 Preliminaries

For simplicity, all languages are over  $\{0, 1\}$ . We assume knowledge of the basics of complexity theory [8]. Here are a few (perhaps) non-standard notions we use. For a function  $s : \mathbb{N} \rightarrow \mathbb{N}$ ,  $\text{poly}(s(n))$  is shorthand for  $O(s(n)^c)$  for some constant  $c$ , and  $\tilde{O}(s(n))$  is shorthand for  $s(n) \cdot \text{poly}(\log n)$ . Define  $\text{SIZE}(s(n))$  to be the class of languages computable by a circuit family of size  $O(s(n))$ . Define  $\Sigma_2\text{TIME}[t(n)]$  to be the class of languages recognizable by a  $\Sigma_2$  machine in time  $O(t(n))$ ; more precisely, the languages  $L$  such that there exists a linear

time machine  $M$  such that for all strings  $x$ ,

$$x \in L \iff (\exists y \in \{0, 1\}^{t(|x|)})(\forall z \in \{0, 1\}^{t(|x|)})[M(x, y, z) \text{ accepts}].$$

In some of our results, we apply the well-known PARITY lower bound of Håstad:

► **Theorem 2.1** (Håstad [12]). *For every  $k \geq 2$ , PARITY cannot be computed by circuits with AND, OR, and NOT gates of depth  $k$  and size  $2^{o(n^{1/(k-1)})}$ .*

### Machine model

The machine model used in our results may be any model with random access to the input via addressing, such as a random-access Turing machine. The main component we want is that the “address” of the bit/symbol/word being read at any step is stored as a readable and writable binary integer.

### A remark on sub-polynomial reductions

In Definition 1.1 we defined sub-polynomial time reductions to output out-of-bounds characters which denote the end of an output string. We could also have defined our reductions to output a string of length  $2^{\lceil c \log_2 n \rceil}$  on an input of length  $n$ , for some fixed constant  $c \geq 1$ . This makes it easy for the reduction to know the “end” of the output. We can still compute the length  $\ell$  of the output in  $O(\log \ell)$  time via Definition 1.1, by performing a doubling search on the indices  $i$  to find one  $\star$  (trying the indices 1, 2, 4, 8, etc.), then performing a binary search for the first  $\star$ . The results in this paper hold for either reduction model (but the encoding of MCSP may have to vary in trivial ways, depending on the reduction notion used).

### Encoding MCSP

Let  $y_1, \dots, y_{2^\ell} \in \{0, 1\}^k$  be the list of  $k$ -bit strings in lex order. Given  $f : \{0, 1\}^\ell \rightarrow \{0, 1\}$ , the truth table of  $f$  is defined to be  $tt(f) := f(y_1)f(y_2) \cdots f(y_{2^\ell})$ .

The truth table of a circuit is the truth table of the function it computes. Let  $T \in \{0, 1\}^*$ . The *function encoded by  $T$* , denoted as  $f_T$ , is the function satisfying  $tt(f_T) = T0^{2^k - |T|}$ , where  $k$  is the minimum integer satisfying  $2^k \geq T$ . The *circuit complexity of  $T$* , denoted as  $CC(T)$ , is simply the minimum number of gates of any circuit computing  $f_T$ .

There are several possible encodings of MCSP we could use. The main point we wish to stress is that it’s possible to encode the circuit size parameter  $k$  in essentially unary or in binary, and our results remain the same. (This is important, because some of our proofs superficially seem to rely on a short encoding of  $k$ .) We illustrate our point with two encodings, both of which are suitable for the reduction model of Definition 1.1. First, we may define MCSP to be the set of strings  $Tx$  where  $|T|$  is the largest power of two satisfying  $|T| < |Tx|$  and  $CC(f_T) \leq |x|$ ; we call this a *unary encoding* because  $k$  is effectively encoded in unary. (Note we cannot detect if a string has the form  $1^k$  in logtime, so we shall let any  $k$ -bit string  $x$  denote the parameter  $k$ . Further note that, if the size parameter  $k > |T|/2$ , then the instance would be trivially a yes-instance. Hence this encoding captures the “interesting” instances of the problem.) Second, we may define MCSP to be the set of binary strings  $Tk$  such that  $|T|$  is the largest power of two such that  $|T| < |Tk|$ ,  $k$  is written in binary (with most significant bit 1) and  $CC(f_T) \leq k$ . Call this the *binary encoding*.

► **Proposition 2.** *There are TIME(poly(log  $n$ )) reductions between the unary encoding of MCSP and the binary encoding of MCSP.*

The proof is a simple exercise, in Appendix A. More points on encoding MCSP for these reductions can be found there as well.

Another variant of MCSP has the size parameter fixed to a large value; this version has been studied extensively in the context of KT-complexity [2, 4]. Define  $\text{MCSP}'$  to be the version with circuit size parameter set to  $|T|^{1/2}$ , that is,  $\text{MCSP}' := \{T \mid CC(T) \leq |T|^{1/2}\}$ . To the best of our knowledge, all theorems in this paper hold for  $\text{MCSP}'$  as well; indeed most of the proofs only become simpler for this case.

### A simple lemma on the circuit complexity of substrings

We also use the fact that for any string  $T$ , the circuit complexity of an arbitrary substring of  $T$  can be bounded via the circuit complexity of  $T$ .

► **Lemma 2.2** ([19]). *There is a universal  $c \geq 1$  such that for any binary string  $T$  and any substring  $S$  of  $T$ ,  $CC(f_S) \leq CC(f_T) + c \log |T|$ .*

**Proof.** Let  $c'$  be sufficiently large in the following. Let  $k$  be the minimum integer satisfying  $2^k \geq |T|$ , so the Boolean function  $f_T$  representing  $T$  has truth table  $T0^{2^k - |T|}$ . Suppose  $C$  is a size- $s$  circuit for  $f_T$ . Let  $S$  be a substring of  $T = t_1 \cdots t_{2^k} \in \{0, 1\}^{2^k}$ , and let  $A, B \in \{1, \dots, 2^k\}$  be such that  $S = t_A \cdots t_B$ . Let  $\ell \leq k$  be a minimum integer which satisfies  $2^\ell \geq B - A$ . We wish to construct a small circuit  $D$  with  $\ell$  inputs and truth table  $S0^{2^\ell - (B - A)}$ . Let  $x_1, \dots, x_{2^\ell}$  be the  $\ell$ -bit strings in lex order. Our circuit  $D$  on input  $x_i$  first computes  $i + A$ ; if  $i + A \leq B - A$  then  $D$  outputs  $C(x_{i+A})$ , otherwise  $D$  outputs 0. Note there are circuits of  $c' \cdot n$  size for addition of two  $n$ -bit numbers (this is folklore). Therefore in size at most  $c' \cdot k$  we can, given input  $x_i$  of length  $\ell$ , output  $i + A$ . Determining if  $i + A \leq B - A$  can be done with  $(c' \cdot \ell)$ -size circuits. Therefore  $D$  can either be implemented as a circuit of size at most  $s + c'(k + \ell + 1)$ . To complete the proof, let  $c \geq 3c'$ . ◀

## 3 MCSP and Sub-Polynomial Time Reductions

In this section, we prove the following impossibility results for NP-hardness of MCSP:

**Reminder of Theorem 1.2.** *For every  $\delta < 1/2$ , there is no  $\text{TIME}(n^\delta)$  reduction from PARITY to MCSP. As a corollary, MCSP is not  $\text{AC0}[2]$ -hard under  $\text{TIME}(n^\delta)$  reductions.*

The proof has the following outline. First we show that there are  $\text{poly}(\log n)$ -time reductions from PARITY to itself which can “insert  $\text{poly}(n)$  zeroes” into a PARITY instance. Then, assuming there is a  $\text{TIME}(n^\delta)$  reduction from PARITY to MCSP, we use the aforementioned zero-inserting algorithm to turn the reduction into a “natural reduction” (in the sense of Kabanets and Cai [15]) from PARITY to MCSP, where the circuit size parameter  $k$  output by the reduction depends only on the input length  $n$ . Next, we show how to bound the value of  $k$  from above by  $\tilde{O}(n^\delta)$ , by exploiting naturalness. Then we use this bound on  $k$  to construct a  $\Sigma_2$  algorithm for PARITY which existentially guesses an  $\tilde{O}(n^\delta)$ -size circuit for the truth table produced by the reduction, then universally verifies the circuit is correct on all bits of the truth table. Finally, we convert the  $\Sigma_2$  algorithm into a depth-three circuit family of  $2^{\tilde{O}(n^\delta)}$  size, and appeal to Håstad’s  $\text{AC0}$  lower bound for PARITY for a contradiction.

We start with a simple  $\text{poly}(\log n)$ -time reduction for padding a string with zeroes in a  $\text{poly}(n)$ -size set of prescribed bit positions. Let  $S \in \mathbb{Z}^\ell$  for a positive integer  $\ell$ . We say  $S$  is *sorted* if  $S[i] < S[i + 1]$  for all  $i = 1, \dots, \ell - 1$ .

► **Proposition 3.** *Let  $p(n)$  be a polynomial. There is an algorithm  $A$  which, given  $x$  of length  $n$ , a sorted tuple  $S = (i_1, \dots, i_{p(n)})$  of indices from  $\{1, \dots, n + p(n)\}$ , and a bit index  $j = 1, \dots, p(n) + n$ ,  $A(x, S, j)$  outputs the  $j$ th bit of the string  $x'$  obtained by inserting zeroes in the bit positions  $i_1, i_2, \dots, i_{p(n)}$  of  $x$ . Furthermore,  $A(x, S, j)$  runs in  $O(\log^2 n)$  time on  $x$  of length  $n$ .*

**Proof.** Given  $x$  of length  $n$ , a sorted  $S = (i_1, \dots, i_p) \in \{1, \dots, n + p\}^p$ , and an index  $j = 1, \dots, n + p$ , first  $A$  checks if  $j \in S$  in  $O(\log^2 n)$  time by binary search, comparing pairs of  $O(\log n)$ -bit integers in  $O(\log n)$  time. If yes, then  $A$  outputs 0. If no, there are two cases: either (1)  $j < i_1$ , or (2)  $i_k < j$  for some  $k = 1, \dots, p$ . In case (1),  $A$  simply outputs  $x_j$ . In case (2),  $A$  outputs  $x_{j-k}$ . (Note that computing  $j - k$  is possible in  $O(\log n)$  time.) It is easy to verify that the concatenation of all outputs of  $A$  over  $j = 1, \dots, |x| + p$  is the string  $x$  but with zeroes inserted in the bit positions  $i_1, \dots, i_p$ . ◀

Let  $t(n) = n^{1-\varepsilon}$  for some  $\varepsilon > 0$ . The next step is to show that a  $\text{TIME}(t(n))$  reduction from  $\text{PARITY}$  to  $\text{MCSP}$  can be turned into a *natural* reduction, in the following sense:

► **Definition 3.1** (Kabanets-Cai [15]). A reduction from a language  $L$  to  $\text{MCSP}$  is *natural* if the size of all output instances and the size parameters  $k$  depend only on the length of the input to the reduction.

The main restriction in the above definition is that the size parameter  $k$  output by the reduction does not vary over different inputs of length  $n$ .

► **Claim 1.** *If there is a  $\text{TIME}(t(n))$  reduction from  $\text{PARITY}$  to  $\text{MCSP}$ , then there is a  $\text{TIME}(t(n) \log^2 n)$  natural reduction from  $\text{PARITY}$  to  $\text{MCSP}$ . Furthermore, the value of  $k$  in this natural reduction is  $\tilde{O}(t(n))$ .*

**Proof.** By assumption, we can choose  $n$  large enough to satisfy  $t(2n) \log(2n) \ll n$ . We define a new (natural) reduction  $R'$  from  $\text{PARITY}$  to  $\text{MCSP}$ :

$R'(x, i)$  begins by gathering a list of the bits of the input that affect the size parameter  $k$  of the output, for a hypothetical  $2n$ -bit input which has zeroes in the positions read by  $R$ . This works as follows. We simulate the  $\text{TIME}(t(n))$  reduction  $R$  from  $L$  to  $\text{MCSP}$  on the output indices corresponding to bits of the size parameter  $k$ , as if  $R$  is reading an input  $x'$  of length  $2n$ . When  $R$  attempts to read a bit of the input, record the index  $i_j$  requested in a list  $S$ , and continue the simulation as if the bit at position  $i_j$  is a 0. Since the  $\text{MCSP}$  instance is polynomial in size,  $k$  written in binary is at most  $O(\log n)$  bits (otherwise we may simply output a trivial “yes” instance), so the number of indices of the output that describe  $k$  is at most  $O(\log n)$  in the binary encoding. It follows that the size parameter  $k$  in the output depends on at most  $t(2n) \log(2n)$  bits of the (hypothetical)  $2n$ -bit input. Therefore  $|S| \leq t(2n) \log(2n)$ . Sort  $S = (i_1, \dots, i_{|S|})$  in  $O(t(n) \log^2 n)$  time, and remove duplicate indices.

$R'$  then simulates the  $\text{TIME}(t(n))$  reduction  $R(x, i)$  from  $\text{PARITY}$  to  $\text{MCSP}$ . However, whenever an input bit  $j$  of  $x$  is requested by  $R$ , if  $j \leq n + |S|$  then run the algorithm  $A(x, S, j)$  from Proposition 3 to instead obtain the  $j$ th bit of the  $O(n + |S|)$ -bit string  $x'$  which has zeroes in the bit positions in the sorted tuple  $S$ . Otherwise, if  $j > n + |S|$  and  $j \leq 2n$  then output 0, and if  $j > 2n$  then output  $\star$  (out of bounds). Since the algorithm of Proposition 3 runs in  $O(\log^2 n)$  time, this step of the reduction takes  $O(t(n) \log^2 n)$  time.



That is, the reduction  $R'$  first looks for all the bits in a  $2n$ -bit input that affect the output size parameter  $k$  in the reduction  $R$ , assuming the bits read are all 0. Then  $R'$  runs  $R$  on a simulated string  $2n$ -bit string  $x'$  for which all those bits are zero (and possibly more at the end, to enforce  $|x'| = 2n$ ). Since the parity of  $x'$  equals the parity of  $x$ , the MCSP instance output by  $R'$  is a yes-instance if and only if  $x$  has odd parity. However for the reduction  $R'$ , the output parameter  $k$  is now a function of only the input length; that is,  $R'$  is natural.

Now let us argue for an upper bound on  $k$ . Define a function  $f(i)$  which computes  $z := 0^n$ , then runs and outputs  $R'(z, i)$ . The truth table of  $f$ ,  $tt(f)$ , is therefore an instance of MCSP. Since  $R'$  is natural, the value of  $k$  appearing in  $tt(f)$  is the *same* as the value of  $k$  for all length- $n$  instances of PARITY.

However, the circuit complexity of  $f$  is *small*: on any  $i$ ,  $R'(0^n, i)$  can be computed in time  $O(t(n) \log^2 n)$ . Therefore the circuit complexity of  $f$  is at most some  $s$  which is  $\tilde{O}(t(n))$ . In particular, the  $\text{TIME}(t(n) \log^2 n)$  reduction can be efficiently converted to a circuit, with any bit of the input  $0^n$  efficiently computed in  $O(\log n)$  time at every request (the only thing to check is that the index requested doesn't exceed  $n$ ). As the instance  $f$  of MCSP has  $CC(f) \leq s$ , by Lemma 2.2 the truth table  $T$  in the instance  $tt(f)$  has  $CC(T) \leq cs$  as well for some constant  $c$ .

Since  $0^n$  has even parity, the truth table of  $f$  is not in MCSP. This implies that the value of  $k$  in the instance  $tt(f)$  must be *less than*  $cs = \tilde{O}(t(n))$ . Therefore the value of  $k$  fixed in the reduction from PARITY to MCSP must be at most  $\tilde{O}(t(n) \log^2 n)$ . ◀

Now, we show that efficient reductions from PARITY to MCSP yield efficient  $\Sigma_2$  algorithms for PARITY:

► **Claim 2.** *If there is a  $\text{TIME}(t(n))$  reduction from PARITY to MCSP, then there is a  $\Sigma_2 \text{TIME}(\tilde{O}(t(n)))$  algorithm for PARITY.*

**Proof.** Construct a  $\Sigma_2$  algorithm for PARITY as follows:

Given an input  $x$ , existentially guess a circuit  $C$  with  $O(\log n)$  inputs and size at most  $s = \tilde{O}(t(n))$ , where  $s$  is taken from Claim 1. Then universally verify over all possible  $O(\log n)$ -bit inputs  $i$  to  $C$  that  $C(i) = R'(x, i)$ , where  $R'$  is from Claim 1. If yes, then *accept*, else *reject*.

Since we know the value of the size parameter in the instance output by  $R'(x, \cdot)$  is at most  $s$  (from Claim 1), there is a circuit  $C$  of size at most  $s$  with the above property if and only if  $x$  has odd parity. Since the number of inputs to  $C$  is  $O(\log n)$ , the universal quantification in the above procedure is only  $O(\log n)$  bits. Verification also takes  $\tilde{O}(t(n))$  time, since  $C$  can be evaluated in  $\tilde{O}(t(n))$  time on any input. Hence the  $\Sigma_2$  procedure has the claimed running time. ◀

Finally, we can complete the proof of Theorem 1.2:

**Proof of Theorem 1.2.** Suppose that PARITY has a  $\text{TIME}(n^\delta)$  reduction from PARITY to MCSP, for some  $\delta < 1/2$ . Then by Claim 2, there is a  $\Sigma_2$  algorithm for PARITY running in  $\tilde{O}(n^\delta)$  time. Such an algorithm can be converted into a depth-three OR-AND-OR circuit of size  $2^{\tilde{O}(n^\delta)}$ : the top OR at the output has incoming wires for all possible  $2^{\tilde{O}(n^\delta)}$  existential guesses for the  $\Sigma_2$  machine, the middle AND tries all  $2^{\tilde{O}(n^\delta)}$  universal guesses, and the remaining deterministic computation on  $\tilde{O}(n^\delta)$  bits is computable with a CNF (AND of ORs) of size  $2^{\tilde{O}(n^\delta)}$ . Therefore, the assumed reduction implies that PARITY has depth-three AC0 circuits of size  $2^{\tilde{O}(n^\delta)}$ . For  $\delta < 1/2$ , this is false by Håstad (Theorem 2.1). ◀

► **Remark.** We used only the following properties of PARITY in the above proof: (a) one can insert zeroes into a string efficiently without affecting its membership in PARITY, (b) PARITY has trivial no-instances (strings of all zeroes), and (c) PARITY lacks small depth-three circuits. We imagine that some of the ideas in the above proof may be useful for other “non-hardness” results in the future.

## 4 NP-Hardness of MCSP Implies Lower Bounds

We now turn to stronger reducibility notions, showing that even NP-hardness of MCSP under these reductions implies separation results that currently appear out of reach.

### 4.1 Consequences of NP-Hardness Under Polytime and Logspace Reductions

Our two main results here are:

**Reminder of Theorem 1.3.** *If MCSP is NP-hard under polynomial-time reductions, then  $\text{EXP} \neq \text{NP} \cap \text{P}_{/poly}$ . Consequently,  $\text{EXP} \neq \text{ZPP}$ .*

**Reminder of Corollary 1.4.** *If MCSP is NP-hard under logarithmic space reductions, then  $\text{PSPACE} \neq \text{ZPP}$ .*

These theorems follow from establishing that the NP-hardness of MCSP and small circuits for EXP implies  $\text{NEXP} = \text{EXP}$ . In fact, it suffices that MCSP is hard for only sparse languages in NP. (Recall that a language  $L$  is *sparse* if there is a  $c$  such that for all  $n$ ,  $|L \cap \{0, 1\}^n| \leq n^c + c$ .)

► **Theorem 4.1.** *If every sparse language in NP has a polynomial-time reduction to MCSP, then  $\text{EXP} \subseteq \text{P}_{/poly} \implies \text{EXP} = \text{NEXP}$ .*

**Proof.** Suppose that MCSP is hard for sparse NP languages under polynomial-time reductions, and that  $\text{EXP} \subseteq \text{P}_{/poly}$ . Let  $L \in \text{NTIME}(2^{n^c})$  for some  $c \geq 1$ . It is enough to show that  $L \in \text{EXP}$ .

Define the padded language  $L' := \{x01^{2^{|x|^c}} \mid x \in L\}$ . The language  $L'$  is then a sparse language in NP. By assumption, there is a polynomial time reduction from  $L'$  to MCSP. Composing the obvious reduction from  $L$  to  $L'$  with the reduction from  $L'$  to MCSP, we have a  $2^{c \cdot n^c}$ -time reduction  $R$  from  $n$ -bit instances of  $L$  to  $2^{c \cdot n^c}$ -bit instances of MCSP, for some constant  $c'$ . Define the language

$$\text{BITS}_R := \{(x, i) \mid \text{the } i\text{th bit of } R(x) \text{ is } 1\}.$$

$\text{BITS}_R$  is clearly in EXP. Since  $\text{EXP} \subseteq \text{P}_{/poly}$ , for some  $d \geq 1$  there is a circuit family  $\{C_n\}$  of size at most  $n^d + d$  computing  $\text{BITS}_R$  on  $n$ -bit inputs.

Now, on a given instance  $x$  of  $L$ , the circuit  $D(i) := C_{2^{|x|+c' \cdot |x|^c}}(x, i)$  has  $c' \cdot |x|^c$  inputs (ranging over all possible  $i = 1, \dots, 2^{c' \cdot |x|^c}$ ) and size at most  $s(|x|) := (2 + c')^d |x|^{cd} + d$ , such that  $tt(D)$  is the output of  $R(x)$ . Therefore, for every  $x$ , the truth tables output by  $R(x)$  all have circuit complexity at most  $e \cdot s(|x|)$  for some constant  $e$ , by Lemma 2.2. This observation leads to the following exponential time algorithm for  $L$ :

On input  $x$ , run the reduction  $R(x)$ , obtaining an exponential sized instance  $\langle T, k \rangle$  of MCSP. If  $k > e \cdot s(|x|)$  then *accept*. Otherwise, cycle through every circuit  $E$  of size at most  $k$ ; if  $tt(E) = T$  then *accept*. If no such  $E$  is found, *reject*.



Producing the truth table  $T$  takes exponential time, and checking all  $2^{O(s(n) \log s(n))}$  circuits of size  $O(s(n))$  on all polynomial sized inputs to the truth table also takes exponential time. As a result  $L \in \text{EXP}$ , which completes the proof.  $\blacktriangleleft$

The same argument can be used to prove collapses for other reducibilities. For example, swapping time for space in the proof of Theorem 4.1, we obtain:

► **Corollary 4.2.** *If MCSP is NP-hard under logspace reductions, then  $\text{PSPACE} \subseteq \text{P}_{/\text{poly}} \implies \text{NEXP} = \text{PSPACE}$ .*

Theorem 4.1 shows that complexity class separations follow from establishing that MCSP is NP-hard in the most general sense. We now prove Theorem 1.3, that NP-hardness of MCSP implies  $\text{EXP} \neq \text{NP} \cap \text{P}_{/\text{poly}}$ :

**Proof of Theorem 1.3.** By contradiction. Suppose MCSP is NP-hard and  $\text{EXP} = \text{NP} \cap \text{P}_{/\text{poly}}$ . Then  $\text{EXP} \subset \text{P}_{/\text{poly}}$  implies  $\text{NEXP} = \text{EXP}$  by Theorem 4.1, but  $\text{NEXP} = \text{EXP} \subseteq \text{NP}$ , contradicting the nondeterministic time hierarchy [20].  $\blacktriangleleft$

Corollary 1.4 immediately follows from the same argument as Theorem 1.3, applying Corollary 4.2.

We would like to strengthen Theorem 1.3 to show that the NP-hardness of MCSP actually implies *circuit* lower bounds such as  $\text{EXP} \not\subseteq \text{P}_{/\text{poly}}$ . This seems like a more natural consequence: an NP-hardness reduction would presumably be able to print truth tables of high circuit complexity from no-instances of low complexity. (Indeed this is the intuition behind Kabanets and Cai’s results concerning “natural” reductions [15].)

## 4.2 Consequences of NP-Hardness under AC0 Reductions

Now we turn to showing consequences of assuming that MCSP is NP-hard under uniform AC0 reductions. Here we obtain consequences so strong that we are skeptical the hypothesis is true.

**Reminder of Theorem 1.5.** *If MCSP is NP-hard under logtime-uniform AC0 reductions, then  $\text{NP} \not\subseteq \text{P}_{/\text{poly}}$  and  $\text{E} \not\subseteq \text{i.o.-SIZE}(2^{\delta n})$  for some  $\delta > 0$ . As a consequence,  $\text{P} = \text{BPP}$  also follows.*

We will handle the two consequences in two separate theorems.

► **Theorem 4.3.** *If MCSP is NP-hard under LOGTIME-uniform AC0 reductions, then  $\text{NP} \subseteq \text{P}_{/\text{poly}} \implies \text{NEXP} \subseteq \text{P}_{/\text{poly}}$ .*

**Proof.** The proof is similar in spirit to that of Theorem 4.1. Suppose that MCSP is NP-hard under LOGTIME-uniform AC0 reductions, and that  $\text{NP} \subseteq \text{P}_{/\text{poly}}$ . Then  $\Sigma_k \text{P} \subseteq \text{P}_{/\text{poly}}$  for every  $k \geq 1$ .

Let  $L \in \text{NEXP}$ ; in particular, let  $L \in \text{NTIME}(2^{n^c})$  for some  $c$ . As in Theorem 4.1, define the sparse NP language  $L' = \{x01^t \mid x \in L, t = 2^{|x|^c}\}$ . By assumption, there is a LOGTIME-uniform AC0 reduction  $R$  from the sparse language  $L'$  to MCSP. This reduction can be naturally viewed as a  $\Sigma_k \text{P}$  reduction  $S(\cdot, \cdot)$  from  $L$  to exponential-sized instances of MCSP, for some constant  $k$ . In particular,  $S(x, i)$  outputs the  $i$ th bit of the reduction  $R$  on input  $x01^t$ , and  $S$  can be implemented in  $\Sigma_k \text{P}$ , and hence in  $\text{P}_{/\text{poly}}$  as well.

That is, for all inputs  $x$ , the string  $S(x, 1) \cdots S(x, 2^{O(|x|^c)})$  is the truth table of a function with  $\text{poly}(|x|)$ -size circuits. Therefore by Lemma 2.2, the truth table of the MCSP instance

being output on  $x$  must have a  $\text{poly}(|x|)$ -size circuit. We can then decide  $L$  in  $\Sigma_{k+2}\text{P}$  time: on an input  $x$ , existentially guess a circuit  $C$  of  $\text{poly}(|x|)$  size, then for all inputs  $y$  to  $C$ , verify that  $S(x, y) = C(y)$ . The latter equality can be checked in  $\Sigma_k\text{P}$ . As a result, we have  $\text{NEXP} \subseteq \Sigma_{k+2}\text{P} \subseteq \text{P}/\text{poly}$ .  $\blacktriangleleft$

► **Theorem 4.4.** *If MCSP is NP-hard under P-uniform AC0 reductions, then there is a  $\delta > 0$  such that  $\text{E} \not\subseteq \text{i.o.-SIZE}(2^{\delta n})$ . As a consequence,  $\text{P} = \text{BPP}$  also follows from the same assumption (Impagliazzo and Wigderson [14]).*

**Proof.** Assume the opposite: that MCSP is NP-hard under P-uniform AC0 reductions and for every  $\epsilon > 0$ ,  $\text{E} \subset \text{i.o.-SIZE}(2^{\epsilon n})$ . By Agrawal *et al.* [1] (Theorem 4.1), all languages hard for NP under P-uniform AC0 reductions are also hard for NP under P-uniform NC0 reductions. Therefore MCSP is NP-hard under P-uniform NC0 reductions. Since in an NC0 circuit all outputs depend on a constant number of input bits, the circuit size parameter  $k$  in the output of the reduction depends on only  $O(\log n)$  input bits. By Claim 1, the NC0 reduction from PARITY to MCSP can be converted into a natural reduction. Therefore we may assume that the size parameter  $k$  in the output of the reduction is a function of only the length of the input to the reduction.

Let  $R$  be a polynomial-time algorithm that on input  $1^n$  produces a P-uniform NC0 circuit  $C_n$  on  $n$  inputs that reduces PARITY to MCSP. Fix  $c$  such that  $R$  runs in at most  $n^c + c$  time and every truth table produced by the reduction is of length at most  $n^c + c$ . Define an algorithm  $R'$  as follows:

On input  $(n, i, b)$ , where  $n$  is a binary integer,  $i = 1, \dots, n^c + c$ , and  $b \in \{0, 1\}$ , run  $R(1^n)$  to produce the circuit  $C_n$ , then evaluate  $C_n(0^n)$  to produce a truth table  $T_n$ . If  $b = 0$ , output the  $i^{\text{th}}$  bit of  $C_n$ . If  $b = 1$ , output the  $i^{\text{th}}$  bit of  $T_n$ .

For an input  $(n, i, b)$ ,  $R'$  runs in time  $O(n^c)$ ; when  $m = |(n, i, b)|$ , this running time is  $2^{O(m)} \leq n^{O(1)}$ . By assumption, for every  $\epsilon > 0$ ,  $R'$  has circuits  $\{D_m\}$  of size  $O(2^{\epsilon m}) \leq O(n^{2\epsilon})$  for infinitely many input lengths  $m$ . This has two important consequences:

1. For every  $\epsilon > 0$  there are infinitely many input lengths  $m = O(\log n)$  such that the size parameter  $k$  in the natural reduction from PARITY to MCSP is at most  $n^{2\epsilon}$  (or, the instance is trivial). To see this, first observe that  $0^n$  is always a no-instance of PARITY, so  $R(0^n)$  always maps to a truth table  $T_m$  of circuit complexity greater than  $k(n)$  (for some  $k(n)$ ). Since  $R'(n, i, 1)$  prints the  $i^{\text{th}}$  bit of  $R(0^n)$ , and the function  $R'(n, \cdot, 1)$  is computable with an  $O(n^{2\epsilon})$ -size circuit  $D_n$ , the circuit complexity of  $T_m$  is at most  $O(n^{2\epsilon})$ , by Lemma 2.2. Therefore the output size parameter  $k$  of  $R(0^n)$  for these input lengths  $m$  is at most  $O(n^{2\epsilon})$ .
2. On the *same* input lengths  $m$  for which  $k$  is  $O(n^{2\epsilon})$ , the *same* circuit  $D_m$  of size  $O(n^{2\epsilon})$  can compute any bit of the NC0 circuit  $C_n$  that reduces PARITY to MCSP. This follows from simply setting  $b = 0$  in the input of  $D_m$ .

The key point is that both conditions are simultaneously satisfied for infinitely many input lengths  $m$ , because both computations are made by the same  $2^{O(n)}$  time algorithm  $R'$ . We use these facts to construct an  $\text{i.o.-}\Sigma_2\text{TIME}(\tilde{O}(n^{2\epsilon}))$  algorithm  $A$ , as follows:

On input  $(x, D)$ , where  $n = |x|$  and  $D$  is an  $O(n^{2\epsilon})$  size circuit with  $m = O(\log n)$  inputs:

Assume  $D$  computes  $R'(n, i, b)$  on inputs such that  $m = |(n, i, b)|$ . Evaluate  $D$  on  $n$ ,  $O(\log n)$  different choices of  $i$ , and  $b = 0$ , to construct the portion of the NC0 circuit

$C_n$  that computes the size parameter  $k$  in the output of the reduction from PARITY to MCSP. Then, use this  $O(\log n)$ -size subcircuit to compute the value of  $k$  for the input length  $n$ .

Next, nondeterministically guess a circuit  $C'$  of size at most  $k$ ; we wish to verify that for all  $i$ ,  $C'(i)$  outputs the  $i^{\text{th}}$  bit of the truth table  $T_x$  produced by the NCO reduction on input  $x$ . We can verify this by noting that the  $i^{\text{th}}$  bit of  $T_x$  can also be computed in  $O(n^\epsilon)$  time, via  $D$ . Namely, universally choose an  $i$ , and produce the  $O(1)$ -size subcircuit that computes the  $i^{\text{th}}$  output bit of the NCO circuit  $C_n$  (by making a constant number of queries to  $D$  with  $b = 0$ ). Then, simulate the resulting  $O(1)$ -size subcircuit on the relevant input bits of  $x$  to compute the  $i^{\text{th}}$  bit of  $T_x$ , and check that  $C'(i)$  equals this bit. If all checks pass, *accept*, else *reject*.

Assuming the circuit  $D$  actually computes  $R'$ ,  $A(x, D)$  computes PARITY correctly. For every  $\epsilon > 0$ , there are infinitely many  $m$  such that the circuit size parameter  $k$  is at most  $2^{O(\epsilon m)} \leq O(n^{2\epsilon})$ , and the circuit  $D$  of size  $2^{O(\epsilon m)} \leq O(n^{2\epsilon})$  exists. Under these conditions, the above  $\Sigma_2$  algorithm  $A$  runs in  $\tilde{O}(n^{2\epsilon})$  time. As a result, for every  $\epsilon > 0$  we can find for infinitely many  $n$  such that the algorithm  $A$  has a corresponding depth-3 AC0 circuit of  $2^{\tilde{O}(n^\epsilon)}$  size. Suppose we hardwire the  $O(n^{2\epsilon})$ -size circuit  $D$  that computes  $R'$  into the corresponding AC0 circuit, on input lengths  $n$  for which  $D$  exists. Then for all  $\epsilon > 0$ , PARITY can be solved infinitely often with depth-3 AC0 circuits of  $2^{\tilde{O}(n^\epsilon)}$  size, contradicting Håstad (Theorem 2.1). ◀

**Proof of Theorem 1.5.** Suppose MCSP is NP-hard under logtime-uniform AC0 reductions. The consequence  $\text{E} \not\subseteq \text{SIZE}(2^{\delta n})$  was already established in Theorem 4.4. The consequence  $\text{NP} \not\subseteq \text{P}_{/poly}$  follows immediately from combining Theorem 4.3 and Theorem 4.4. ◀

### 4.3 The Hardness of Nondeterministic Circuit Complexity

Finally, we consider the generalization of MCSP to the nondeterministic circuit model. Recall that a *nondeterministic circuit*  $C$  of size  $s$  takes two inputs, a string  $x$  on  $n$  bits and a string  $y$  on at most  $s$  bits. We say  $C$  computes a function  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  if for all  $x \in \{0, 1\}^n$ ,  $f(x) = 1 \iff$  there is a  $y$  of length at most  $s$  such that  $C(x, y) = 1$ .

Observe that the Cook-Levin theorem implies that every nondeterministic circuit  $C$  of size  $s$  has an equivalent nondeterministic depth-two circuit  $C'$  of size  $s \cdot \text{poly}(\log s)$  (and *unbounded* fan-in). Therefore, it is no real loss of generality to define *Nondeterministic MCSP* (NMCSP) to be the set of all pairs  $\langle T, k \rangle$  where  $T$  is the truth table of a function computed by a depth-two nondeterministic circuit of size at most  $k$ . This problem is in  $\Sigma_2 P$ : given  $\langle T, k \rangle$ , existentially guess a nondeterministic circuit  $C$  of size at most  $k$ , then for every  $x$  such that  $T(x) = 1$ , existentially guess a  $y$  such that  $C(x, y) = 1$ ; for every  $x$  such that  $T(x) = 0$ , universally verify that for all  $y$ ,  $C(x, y) = 0$ . However, NMCSP is not known to be  $\Sigma_2 P$ -hard. (The proof of Theorem 1.6 below will work for the depth-two version with unbounded fan-in, and the unrestricted version.)

Recall that it is *known* that MCSP for depth-two circuits is NP-hard [10, 6]. That is, the “deterministic counterpart” of MCSP is known to be NP-hard.

**Reminder of Theorem 1.6.** *If NMCSP is MA-hard under polynomial-time reductions, then  $\text{EXP} \not\subseteq \text{P}_{/poly}$ .*

**Proof.** Suppose that NMCSP is MA-hard under polynomial-time reductions, and suppose that  $\text{EXP} \subseteq \text{P}_{/poly}$ . We wish to establish a contradiction. The proof is similar in structure to

other theorems of this section (such as Theorem 4.1). Let  $L \in \text{MATIME}(2^{n^c})$ , and define  $L' = \{x01^{2^{|x|^c}} \mid x \in L\} \in \text{MA}$ . By assumption, there is a reduction  $R$  from  $L'$  to NMCSPP that runs in polynomial time. Therefore for some constant  $d$  we have a reduction  $R'$  from  $L$  that runs in  $2^{dn^c}$  time, and outputs  $2^{dn^c}$ -sized instances of NMCSPP with a size parameter  $s(x)$  on input  $x$ . Since  $R'$  runs in exponential time, and we assume  $\text{EXP}$  is in  $\text{P}_{/\text{poly}}$ , there is a  $k$  such that for all  $x$ , there is a nondeterministic circuit  $C((x, i), y)$  of  $\leq n^k + k$  size that computes the  $i$ th bit of  $R'(x)$ . Therefore we know that  $s(x) \leq |x|^k + k$  on such instances (otherwise we can trivially *accept*). We claim that  $L \in \text{EXP}$ , by the following algorithm:

Given  $x$ , run  $R'(x)$  to compute  $s(x)$ . If  $s(x) > |x|^k + k$  then *accept*.

For all circuits  $C$  in increasing order of size up to  $s(x)$ ,

Initialize a table  $T$  of  $2^{dn^c}$  bits to be all-zero.

For all  $i = 1, \dots, 2^{dn^c}$  and all  $2^{s(x)}$  possible nondeterministic strings  $y$ ,

Check for each  $i$  if there is a  $y$  such that  $C((x, i), y) = 1$ ; if so, set  $T[i] = 1$ .

If  $T = R'(x)$  then *accept*.

*Reject* (no nondeterministic circuit of size at most  $s(x)$  was found).

Because  $s(x) \leq |x|^k + k$ , the above algorithm runs in  $2^{n^k \cdot \text{poly}(\log n)}$  time and decides  $L$ . Therefore  $L \in \text{EXP}$ . But this implies that  $\text{MAEXP} = \text{EXP} \subseteq \text{P}_{/\text{poly}}$ , which contradicts the circuit lower bound of Buhrman, Fortnow, and Thierauf [9]. ◀

## 5 Conclusion

We have demonstrated several formal reasons why it has been difficult to prove that MCSPP is NP-hard. In some cases, proving NP-hardness would imply longstanding complexity class separations; in other cases, it is simply impossible to prove NP-hardness.

There are many open questions left to explore. Based on our study, we conjecture that:

- If MCSPP is NP-hard under polynomial-time reductions then  $\text{EXP} \not\subseteq \text{P}_{/\text{poly}}$ . We showed that if MCSPP is hard for sparse NP languages then  $\text{EXP} \neq \text{ZPP}$ ; surely a reduction from SAT to MCSPP would provide a stronger consequence.
- MCSPP is (unconditionally) not NP-hard under logtime-uniform AC0 reductions. Theorem 1.2 already implies that MCSPP isn't NP-hard under polylogtime-uniform NC0 reductions. Perhaps this next step isn't far away, since we already know that hardness under P-uniform AC0 reductions implies hardness under P-uniform NC0 reductions (by Agrawal *et al.* [1]).

It seems that we can prove that finding the minimum DNF for a given truth table is NP-hard, because of  $2^{\Omega(n)}$  size lower bounds against DNFs [6]. Since there are  $2^{\Omega(n^\delta)}$  size lower bounds against AC0, can it be proved that finding the minimum AC0 circuit for a given truth table is QuasiNP-hard? In general, can circuit lower bounds imply hardness results for circuit minimization?

**Acknowledgements.** We thank Greg Bodwin and Brynmor Chapman for their thoughtful discussions on these results. We also thank Eric Allender and Oded Goldreich for helpful comments. We're also grateful to Eric for providing a preprint of his work with Dhiraj Holden.

---

**References**

---

- 1 Maindra Agrawal, Eric Allender, Russell Impagliazzo, Toniann Pitassi, and Steven Rudich. Reducing the complexity of reductions. *Computational Complexity*, 10(2):117–138, 2001.
- 2 Eric Allender. When worlds collide: Derandomization, lower bounds, and Kolmogorov complexity. In *FSTTCS*, volume 2245 of *LNCS*, pages 1–15. Springer, 2001.
- 3 Eric Allender. Personal communication, 2014.
- 4 Eric Allender, Harry Buhrman, Michal Koucký, Dieter van Melkebeek, and Detlef Ronneburger. Power from random strings. *SIAM J. Comput.*, 35(6):1467–1493, 2006.
- 5 Eric Allender and Bireswar Das. Zero knowledge and circuit minimization. In *Mathematical Foundations of Computer Science (MFCS), Part II*, pages 25–32, 2014.
- 6 Eric Allender, Lisa Hellerstein, Paul McCabe, Toniann Pitassi, and Michael Saks. Minimizing disjunctive normal form formulas and  $AC^0$  circuits given a truth table. *SIAM J. Comput.*, 38(1):63–84, 2008.
- 7 Eric Allender, Dhiraj Holden, and Valentine Kabanets. The minimum oracle circuit size problem. In *STACS*, pages 21–33, 2015.
- 8 Sanjeev Arora and Boaz Barak. *Computational Complexity - A Modern Approach*. Cambridge University Press, 2009.
- 9 Harry Buhrman, Lance Fortnow, and Thomas Thierauf. Nonrelativizing separations. In *Proceedings of 13th Annual IEEE Conf. Computational Complexity*, pages 8–12, 1998.
- 10 Sebastian Czort. The complexity of minimizing disjunctive normal form formulas. *Master's Thesis, University of Aarhus*, 1999.
- 11 Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. *J. ACM*, 38(3):691–729, 1991.
- 12 Johan Håstad. Almost optimal lower bounds for small depth circuits. In *Proceedings of the 18th Annual ACM Symposium on Theory of Computing*, pages 6–20, 1986.
- 13 Russell Impagliazzo, Valentine Kabanets, and Avi Wigderson. In search of an easy witness: Exponential time vs. probabilistic polynomial time. *JCSS*, 65(4):672–694, 2002.
- 14 Russell Impagliazzo and Avi Wigderson.  $P = BPP$  if E requires exponential circuits: Derandomizing the XOR lemma. In *Proceedings of the 29th Annual ACM Symposium on the Theory of Computing*, pages 220–229, 1997.
- 15 Valentine Kabanets and Jin-Yi Cai. Circuit minimization problem. In *STOC*, pages 73–79, 2000.
- 16 Christos H. Papadimitriou and Mihalis Yannakakis. A note on succinct representations of graphs. *Information and Control*, 71(3):181–185, 1986.
- 17 Sven Skyum and Leslie G Valiant. A complexity theory based on boolean algebra. *J. ACM*, 32(2):484–502, 1985.
- 18 N. V. Vinodchandran. Nondeterministic circuit minimization problem and derandomizing Arthur-Merlin games. *International Journal of Foundations of Computer Science*, 16(6):1297–1308, 2005.
- 19 Ryan Williams. Natural proofs versus derandomization. In *STOC*, pages 21–30, 2013.
- 20 Stanislav Žák. A Turing machine time hierarchy. *Theoretical Computer Science*, 26(3):327–333, 1983.

## A

 Appendix: Unary and Binary Encodings of MCSP

We will describe our reductions in the reduction model with “out of bounds” errors (Definition 1.1). In that model, we may define a *unary encoding of MCSP*  $(T, k)$  to be  $Tx$  where  $|T|$  is the largest power of two such that  $|T| \leq |Tx|$ ,  $k = |x|$ . This encoding is sensible because we may assume WLOG that  $k \leq 2|T|/\log |T|$ : the size of a minimum circuit for a boolean function on  $n$  inputs is always less than  $2^{n+1}/n$ . Similarly, we defined *MCSP*  $(T, k)$  in the *binary encoding* to simply be  $Tk$  where  $|T|$  is the largest power of two such that  $|T| \leq |Tk|$ .

Note that these encodings may be undesirable if one really wants to allow trivial yes instances in a reduction to MCSP where the size parameter  $k$  is too large for the instance to be interesting, or if one wants to allow  $T$  to have length other than a power of two. For those cases, the following binary encoding works: we can encode the instance  $(T, k)$  as the strings  $T00k'$  such that  $k'$  is  $k$  written “in binary” over the alphabet  $\{01, 11\}$ . There are also  $\text{TIME}(\text{poly}(\log n))$  reductions to and from this encoding to the others above, mainly because  $k'$  has length  $O(\log n)$ .

► **Proposition 4.** *There are  $\text{TIME}(\text{poly}(\log n))$  reductions between the unary encoding of MCSP and the binary encoding of MCSP.*

**Proof.** We can reduce from the binary encoding to the unary encoding as follows. Given an input  $y$ , perform a doubling search (probing positions 1, 2, 4, ...,  $2^\ell$ , etc.) until a  $\star$  character is returned. Letting  $2^\ell < |y|$  be the position of the last bit read, this takes  $O(\log |y|)$  probes to the input. Then we may “parse” the input  $y$  into  $T$  as the first  $2^\ell$  bits, and integer  $k'$  as the remainder. To process the integer  $k'$ , we begin by assuming  $k' = 1$ , then we read in  $\log |y|$  bits past the position  $2^\ell$ , doubling  $k'$  for each bit read and adding 1 when the bit read is 1, until  $k' > |y|$  (in which case we don’t have to read further: the instance is trivially yes) or we read a  $\star$  (in which case we have determined the integer  $k'$ ). Finally, if the bit position  $i$  requested is at most  $2^\ell$ , then we output the identical bit from the input  $Tk$ . If not, we print 1 if  $i < 2^\ell + k + 1$ , and  $\star$  otherwise. The overall output of this reduction is  $T1^{k'}$  where  $k < |T|$ . Since addition of  $O(\log n)$  numbers can be done in  $O(\log n)$  time, the above takes  $\text{poly}(\log n)$  time.

To reduce from the unary encoding to the binary encoding, we perform a doubling search on the input  $y$  as in the previous reduction, to find the largest  $\ell$  such that  $2^\ell < |y|$ . Then we let the first  $2^\ell$  bits be  $T$ , and set the parameter  $k = |y| - 2^\ell - 1$ . (Finding  $|y|$  can be done via binary search in  $O(\log |y|)$  calls to the reduction.) From here, outputting the  $i$ th bit of either  $T$  or  $k$  in the binary encoding is easy, since  $|k| = O(\log |y|)$ .

◀

# Circuits with Medium Fan-In\*

Pavel Hrubeš<sup>1</sup> and Anup Rao<sup>2</sup>

1 Institute of Mathematics of the Czech Academy of Sciences  
Prague, Czech Republic  
pahrubes@gmail.com

2 Department of Computer Science and Engineering, University of Washington  
Seattle, USA  
anuprao@cs.washington.edu

---

## Abstract

We consider boolean circuits in which every gate may compute an arbitrary boolean function of  $k$  other gates, for a parameter  $k$ . We give an explicit function  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  that requires at least  $\Omega(\log^2 n)$  non-input gates when  $k = 2n/3$ . When the circuit is restricted to being layered and depth 2, we prove a lower bound of  $n^{\Omega(1)}$  on the number of non-input gates. When the circuit is a formula with gates of fan-in  $k$ , we give a lower bound  $\Omega(n^2/k \log n)$  on the total number of gates.

Our model is connected to some well known approaches to proving lower bounds in complexity theory. Optimal lower bounds for the Number-On-Forehead model in communication complexity, or for bounded depth circuits in  $AC_0$ , or extractors for varieties over small fields would imply strong lower bounds in our model. On the other hand, new lower bounds for our model would prove new time-space tradeoffs for branching programs and impossibility results for (fan-in 2) circuits with linear size and logarithmic depth. In particular, our lower bound gives a different proof for a known time-space tradeoff for oblivious branching programs.

**1998 ACM Subject Classification** F.1.m [Theory of Computation] Computation by Abstract Devices – Miscellaneous

**Keywords and phrases** Boolean circuit, Complexity, Communication complexity

**Digital Object Identifier** 10.4230/LIPIcs.CCC.2015.381

## 1 Introduction

A boolean circuit is usually defined as a directed acyclic graph where vertices (called gates) have in-degree (called fan-in) at most 2. Every gate with fan-in 0 corresponds to an input variable, and all other gates compute an arbitrary boolean function of the values that feed into them. Sometimes the model is restricted to using gates from the DeMorgan basis (i.e. AND, OR, NOT) gates, but this changes the size of the circuit by at most a constant factor. The circuit computes a function  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  if some gate in the circuit evaluates to  $f$ . A formula is a circuit whose underlying graph is a tree. The depth of the circuit is the length of the longest path in the graph.

Since every algorithm with running time  $T(n)$  can be simulated by circuits of size  $\tilde{O}(T(n))$ , one can hope to prove lower bounds on the time complexity of algorithms by proving lower bounds on circuit size. A super-polynomial lower bound on the circuit size of an NP problem

---

\* This work was partially supported by: ERC grant FEALORA 339691, an Alfred P. Sloan Fellowship, the National Science Foundation under agreement CCF-1016565, an NSF Career award, and by the Binational Science Foundation under agreement 2010089.





would imply that  $P \neq NP$ . However, we know of no explicit function (even outside NP) for which we can prove a super-linear lower bound. In contrast, counting arguments imply that almost every function requires circuits of exponential size.

We study a *stronger* model of circuits. We allow the gates to have fan-in  $k$ , where  $k$  is a parameter that depends on  $n$ , and each gate may compute an arbitrary function of its inputs. Typically, we consider the case where  $k$  is a constant fraction of  $n$ . We write  $C_k(f)$  to denote the minimum number of *non-input* gates required to compute  $f$  in this model.

These circuits are much stronger than the models usually studied in the context of proving lower bounds. Nevertheless, we show that many attempts at proving lower bounds on other models of computation can be seen as attempts to prove new lower bounds in our model. Truly exponential lower bounds for  $AC_0$ , optimal lower bounds for the Number-On-Forehead (or NOF) model of communication, or new extractors for varieties over small fields, would all improve the best lower bounds we know how to prove for  $C_k(f)$ . On the other hand, new lower bounds in our model would lead to lower bounds for branching programs and (fan-in 2) circuits of logarithmic depth. Our Theorem 1 already leads to a different proof of the lower bounds on oblivious branching programs given by Babai, Nisan and Szegedy [3]. We elaborate on these connections in Section 4.

Similar models have been studied in past works. Circuits with arbitrary gates and *arbitrarily large* fan-in have been considered for computing *several* boolean functions simultaneously. If  $n$  boolean functions are being computed, the trivial upper bound uses  $n^2$  wires (edges). Super-linear lower bounds on the number of wires are known for circuits of bounded depth in this scenario [9, 17, 18, 14]. Beame, Koutris and Suciú [5], studied a model of communication where  $p$  processors, each with memory  $n/p^{1-\epsilon}$  attempt to compute with a minimal amount of communication. This model is conceptually related to ours, since each such processor can be thought of as a collection of gates with bounded fan-in. Goldreich and Wigderson [12] investigated multilinear arithmetic circuits where the gates are allowed to compute arbitrary multilinear functions of a bounded number of inputs. None of these results seem to give non-trivial lower bounds on  $C_k(f)$ .

Clearly,  $C_n(f) = 1$ , since  $f$  has only  $n$  variables. However, when the fan-in is restricted, the power of circuits dramatically decreases. A counting argument shows that for almost every  $f$ ,  $C_k(f) > 2^{(n-k)-o(n-k)}$ , which is exponentially large even for  $k$  linear in  $n$ . On the other hand, one can show that  $C_k(f) \leq O((n-k)2^{n-k})$  for every  $f$ . The challenge is to obtain such a lower bound for an explicit function  $f$ . If  $f$  depends on all its inputs, then it is easy to see that  $C_k(f) \geq n/k$ . When  $k$  is linear in  $n$ , this trivial lower bound is just a constant.

Chandra, Furst and Lipton [8] defined the Number-on-Forehead model of communication, which we discuss in detail in Section 2.1. They proved lower bounds on branching programs computing the majority function by giving a reduction to the NOF model. The lower bound for the communication model is obtained via Ramsey style argument and displays a tower-like decay. Their reduction is easily adapted to our model as well, yielding super-constant lower bounds on  $C_{2n/3}(\text{Majority})$ . In our work, we use NOF lower bounds to obtain stronger results. We use a different reduction to show: <sup>1</sup>

► **Theorem 1.** *There exists  $f \in P$  such that for every  $\gamma > 0$  and  $n$  large enough,  $C_{(1-\gamma)n}(f) \geq \Omega(\gamma \log^2 n)$ .*

<sup>1</sup> Abusing notation, we write  $f \in P$  to mean that  $f$  is obtained by restricting a polynomial time computable function to  $n$ -bit inputs.



The proof is reminiscent of the approaches in [16, 1, 3, 7, 6] concerning time-space trade-offs for oblivious branching programs.

Next, we define a quantity which is closely related to  $C_k(f)$ . Let  $C_k^2(f)$  denote the smallest number  $m$  such that there exist boolean functions  $g, f_1, \dots, f_m$  with  $f = g(f_1, \dots, f_m)$ , where every  $f_i$  reads at most  $k$  inputs. We prove:

► **Theorem 2.** *There exist  $f \in \mathcal{P}$ ,  $c > 0$ , such that  $C_{(1-\gamma)n}^2(f) \geq \Omega(n^{c\gamma})$ .*

The proof of Theorem 2 involves ideas inspired by Nechiporuk’s [15] lower bound on boolean formula size. We show (Proposition 4) that  $C_k^2(f) \leq C_k(f) \cdot 2^{C_k(f)}$  for every  $f$ , and hence Theorem 2 implies a lower bound of  $\Omega(\gamma \log n)$  on  $C_{(1-\gamma)n}(f)$ . In fact, the specific  $f$  from Theorem 2 satisfies  $C_{2n/3}(f) \leq O(\log n)$ , showing that  $C_{2n/3}^2$  can be exponentially larger than  $C_{2n/3}$ .

Finally, we observe that Nechiporuk’s original proof can be easily extended to formulas with large fan-in. Write  $L_k(f)$  for the smallest number of leaves in a formula computing  $f$  with fan-in at most  $k$ . Nechiporuk gave an explicit function  $f$  for which  $L_2(f) \geq \Omega(n^2 / \log n)$ . We prove:

► **Theorem 3.** *There exists  $f \in \mathcal{P}$  such that  $L_k(f) = \Omega(n^2 / k \log n)$ .*

Note that for formulas we are counting leaves and not just the non-input gates. Of course, Theorem 3 implies a lower bound of  $\Omega(n^2 / k^2 \log n)$  on the number of non-input gates as well.

The lower bound in Theorem 1 is stronger than stated. Consider circuits where the gates can have arbitrarily large fan-in, but each gate can read at most  $k$  input variables. Define  $C_k^*(f)$  as the smallest number of non-input gates which read *some* input variable in a circuit computing  $f$ . Then  $C_k^*(f) \leq C_k(f)$ . Our lower bound proofs actually give lower bounds on  $C_k^*(f)$ : both Theorem 1 and Proposition 4 work for  $C_k^*$  as well. On the other hand, we always have  $C_k^*(f) \leq n$ . Hence, proving a super-linear lower bound on  $C_k(f)$  requires a technique which fails to work for  $C_k^*(f)$ .

In Section 2, we discuss the quantities  $C_k$  and  $C_k^2$  in greater detail. In Section 3, we give the proofs our lower bounds. In Section 4, we outline the connections between our model and other problems in complexity theory.

## 2 Circuits of medium fan-in

As mentioned in the introduction, counting arguments show that for almost every  $f$ ,  $C_k(f) > 2^{(n-k)-o(n-k)}$ . The bound is exponential even when  $k$  is very close to  $n$ , and super-linear even when  $k < n - 1.1 \log n$ . It becomes sub-linear when  $k > n - \log n$ . One can check that  $C_{n-1.1 \log \log n}(f) = \Omega(\log n)$  for most functions  $f$ .

The trivial upper bound on the quantity  $C_k^2(f)$  is  $n$ . The bound is tight even if  $k$  is very close to  $n$ : there exists an  $f$  for which  $C_{\lfloor n - \log n - 1 \rfloor}^2(f) = n$ . Indeed, the number of choices for the functions  $g, f_1, \dots, f_m$  is at most

$$2^{2^m} \left( \binom{n}{k} 2^{2^k} \right)^m \leq 2^{2^m + m2^k + nm}.$$

In order to realize all  $n$ -variate functions, we must have  $2^m + m2^k + nm \geq 2^n$ . If  $m = n - 1$  and  $k = \lfloor n - \log n - 1 \rfloor$ , the bound is

$$2^{n-1} + (n-1)2^{n-1}/n + n^2 = 2^n(1 - 1/(2n)) + n^2 < 2^n.$$

An exercise would show that  $\ell \leq \log n$  implies  $C_{n-\ell}^2(f) \leq 2^\ell + \ell$ , thus  $C_k^2$  decreases when  $k$  goes above  $n - \log n$ .

The following proposition relates  $C_k(f)$  to  $C_k^2(f)$ .

► **Proposition 4.**  $C_k^2(f) \leq C_k(f) \cdot 2^{C_k(f)}$ .

**Proof.** Let  $u_1, \dots, u_s$  be the non-input gates in a circuit of size  $s = C_k(f)$  where the gate  $u_s$  evaluates to  $f$ . For every  $i \in [s]$  and every  $\sigma : \{u_1, \dots, u_s\} \rightarrow \{0, 1\}$ , we define a function  $f_{i,\sigma}$  that depends on at most  $k$  input variables, as follows.  $f_{i,\sigma}$  reads the input variables that are read by  $u_i$ , and outputs 1 if and only if there exists some setting of the remaining input variables that could result in the evaluation given in  $\sigma$ . Define  $g$  to be the function that reads the outputs of the  $f_{i,\sigma}$ 's and computes  $f$  by finding the unique  $\sigma$  for which  $f_{i,\sigma} = 1$  for every  $i$ . Formally,  $f = \bigvee_{\sigma: \sigma(u_s)=1} \bigwedge_{i \in [s]} f_{i,\sigma}$ . ◀

Proposition 4 together with Theorem 2 already gives an  $\Omega(\log n)$  lower bound on  $C_{2n/3}(f)$ . However, the exponential loss in the transformation means that even an optimal lower bound (of  $n$ ) on depth-2 circuits would give at most a logarithmic lower bound for general circuits. We show in Proposition 5 that the exponential loss is inevitable.

## 2.1 Communication complexity

In the Number-On-Forehead model of communication complexity [8], there are  $p$  parties that are trying to compute a function  $f(x^1, x^2, \dots, x^p)$ , where each  $x^i$  is a  $n/p$ -bit string. The  $i$ 'th party can see every input except  $x^i$ . To evaluate  $f$ , the parties exchange messages (by broadcast), until one of the parties can transmit the value of  $f$  to the others. The complexity of  $f$  is the number of bits the players need to exchange in order to evaluate  $f$ . Every function can be computed with  $n/p$  bits of communication. The strongest lower bounds known are due to Babai, Nisan and Szegedy [3]. They proved that the generalized inner product function defined by

$$\text{GIP}(x^1, \dots, x^p) = \sum_{i=1}^{n/p} \prod_{j=1}^p x_i^j \pmod 2$$

requires  $\Omega(n/2^{2p})$  bits of communication. They also showed that computing the quadratic character on a sum of numbers requires  $\Omega(n/2^p)$  communication.

The most straightforward connection between circuits and the NOF model is the following observation:

*Suppose that a circuit computing  $f(x^1, \dots, x^p)$  has the property that for every gate  $u$  there is some  $i \in [p]$  such that  $u$  reads no variable from  $x^i$ . Then, if the circuit has  $s$  non-input gates, the function  $f$  can be evaluated using  $s$  bits in the NOF model.*

This does not directly imply a circuit lower bound – in a circuit of linear fan-in, gates may access a constant fraction of each of the blocks  $x_i$ . For example, GIP can be computed by a constant size circuit with fan-in  $n/2$  (imagine two gates, one reading the first half of every  $x^i$ , and the other the second half). Nevertheless, this issue can be partially circumvented, as in [8] or in our Theorem 1, where we use the GIP function to obtain  $C_{2n/3}(f) \geq \Omega(\log^2 n)$  for a related function  $f$ . An explicit function requiring  $\Omega(n/p)$  communication in the NOF model would give an explicit function with  $C_{2n/3}(f) \geq \Omega(\sqrt{n})$ .

**3 The lower bounds**

**3.1 The Nechiporuk method applied to  $L_k(f)$**

The proofs of Theorems 2 and 3 are variations of Nechiporuk’s lower bound on formula size, which we now discuss. Given a boolean function  $f$  on  $n$  variables, a subset of its variables  $S$ , and an assignment  $\sigma$  to the variables outside  $S$ , we define the function  $f_\sigma$  be the function obtained by setting the variables outside  $S$  to  $\sigma$ . It is a function in the variables  $S$ . Any such function is called an  $S$ -subfunction of  $f$ . The number of  $S$ -subfunctions of  $f$  is clearly at most  $\min(2^{2^{|S|}}, 2^{n-|S|})$ .

Nechiporuk finds a function  $f$  whose input is partitioned into intervals  $x_1, x_2, \dots, x_r$ , each of size approximately  $\log n$ , such that for every  $i$ ,  $f$  has  $2^{\Omega(n)}$   $\{x_i\}$ -subfunctions. A simple example is the element distinctness function. Divide the  $n$ -bit input into  $r = n/(2 \log n)$  intervals, each of size  $2 \log n$ , and let  $f(x_1, \dots, x_r) := 1$  iff  $x_1, \dots, x_r \in \{0, 1\}^{2 \log n}$  are distinct. Observe that whenever  $\sigma_2, \dots, \sigma_r \in \{0, 1\}^{2 \log n}$  are distinct, then  $f(x_1, \sigma_2, \dots, \sigma_r)$  rejects precisely on the inputs  $\sigma_2, \dots, \sigma_r$ . Hence  $f$  has at least  $\binom{n^2}{r-1} \geq (\frac{n^2}{r-1})^{r-1} = 2^{\Omega(n)}$   $\{x_1\}$ -subfunctions, and likewise for any  $\{x_i\}$ .

We now prove Theorem 3, which is a straightforward extension of Nechiporuk’s argument for  $k = 2$  to general  $k$ . It is however noteworthy that the bound deteriorates only polynomially with  $k$ .

► **Claim 1.** *Let  $S$  be a subset of variables of  $f$ . Assume that  $f$  can be computed by a formula with fan-in  $k$  in which  $m$  leaves correspond to inputs from  $S$ . Then  $f$  has at most  $2^{O(mk)}$   $S$ -subfunctions.*

**Proof.** Given such a formula computing  $f$ , define the tree  $T$  as the union of all paths going from some variable in  $S$  to the output. Assume that the formula is such that  $|T|$  is smallest possible and, without loss of generality,  $k \geq 3$ . Then  $T$  contains no path  $u, v_1, \dots, v_r$  with  $v_1, \dots, v_r$  having in-degree 1 (in  $T$ ) and  $r > 1$ . For then the value of  $v_r$  is determined by the value of  $u$  and a pair of functions  $g_1, g_2$  of inputs from the complement of  $S$ . We can replace  $v_r$  in our formula by a single gate of fan-in 3, which takes as input  $u$  and two gates computing  $g_1$  and  $g_2$ . This may increase the size of the formula, but decreases the size of  $T$ .

The tree  $T$  has  $m$  leaves. Since there are no edges connecting gates of in-degree 1 in  $T$ , it has at most  $4m$  nodes. Every  $S$ -subfunction can be described using  $4mk$  bits as follows. For each gate  $v$  in  $T$ , it is enough to specify the inputs to  $v$  coming from outside of  $T$ . Since the fan-in of every gate is at most  $k$ , there will be at most  $4mk$  such inputs. Thus  $f$  has at most  $2^{4mk}$   $S$ -subfunctions. ◀

Applying the claim to the element distinctness function, we obtain that every formula computing  $f$  contains  $\Omega(n/k)$  leaves labelled with a variable from  $x_i$ , for every  $i \in \{1, \dots, n/(2 \log n)\}$ . This means that any such formula contains  $\Omega(\frac{n^2}{k \log n})$  leaves altogether.

**3.2 Proof of Theorem 2**

In order to prove our theorem, we will find a function  $f$  that has a stronger property with regards to its subfunctions. Namely,  $f$  will have many  $S$  subfunctions not just for  $S$  coming from a fixed partition of the inputs; it will have many  $S$ -subfunctions for *almost every*  $\log n$ -element set  $S$ .

We define our hard function as follows.  $f(x, y)$  will take as inputs  $x \in \{0, 1\}^{\ell + \log \ell}$  and a  $O(\log^2 \ell)$ -bit string  $y$ . Thus  $f$  is a function of  $n = \ell + O(\log^2 \ell)$  bits in total. We view  $y$  as

representing a subset  $S_y \subset [\ell + \log \ell]$  of  $\log \ell$  variables from  $x$ . Let  $x_{S_y}$  be the projection of  $x$  to the variables in  $S_y$ . We view the  $\log \ell$ -bit string  $x_{S_y}$  as an element of  $[\ell]$ . Let  $S_y^c$  denote the complement of  $S_y$ . Then define the function  $f(x, y)$  to output the  $x_{S_y}$ 'th bit of  $x_{S_y^c}$ .

Given a fixed  $y$ , each setting of  $x_{S_y^c}$  gives a distinct  $S_y$ -subfunction of  $f(x, y)$ . Thus,

► **Claim 2.** *For every  $\log \ell$ -element subset  $S$  of the variables  $x$ ,  $f$  has  $2^\ell$   $S$ -subfunctions.*

To prove Theorem 2, it will be enough to show that any small circuit gives an upper bound on the number of  $S$ -subfunctions of  $f$ , for some  $\log \ell$  element subset  $S$  of the variables in  $x$ .

Suppose that  $f = g(f_1, \dots, f_m)$ , where every  $f_i$  reads at most  $(1 - \gamma)n$  variables. First we observe:

► **Claim 3.** *There exists  $0 < c < 1/2$  such that for every  $0 < \gamma < 1$ , if  $\ell > 100$  and  $m < \ell^{c\gamma}/2$ , then there exists a  $\log \ell$ -element subset  $S$  of the variables  $x$  such that each  $f_i$  reads at most  $(1 - \gamma/2) \log \ell$  of the variables from  $S$ .*

**Proof.** Pick  $\log \ell$  variables  $a_1, \dots, a_{\log \ell}$  from  $x, y$  uniformly at random. With high probability, they will be distinct and they will completely miss the variables  $y$ ; the probability being larger than  $1/2$  if  $\ell > 100$ . For a given  $i$ , let  $X$  be the random variable that counts the number of variables of  $S$  that are read by the gate  $f_i$ . The Chernoff-Hoeffding bound gives,

$$\Pr \left[ \frac{X}{\log \ell} \geq 1 - \gamma/2 \right] \leq e^{-D(1-\gamma/2||1-\gamma) \log \ell} < \ell^{-c\gamma},$$

for a suitable  $c > 0$ . Here,  $D(1 - \gamma/2 || 1 - \gamma) = \gamma/2 \ln(1/2) + (1 - \gamma/2) \ln((1 - \gamma/2)/(1 - \gamma))$  is the Kullback-Leibler divergence. As  $\gamma$  approaches 0, the divergence becomes roughly  $\gamma/2 \ln(1/2) + \gamma/2 > 0.15\gamma$ ; as  $\gamma$  approaches 1 it goes to infinity. Hence we indeed have  $D(1 - \gamma/2 || 1 - \gamma) > c'\gamma$  for some constant  $c' > 0$  and every  $\gamma \in (0, 1)$ , and we can set  $c := (\log_2 e)c'$  (the assumption  $c < 1/2$  is without loss of generality). If  $m < \ell^{c\gamma}/2$ , the union bound gives that there is a  $\log \ell$ -element set  $S$  with the required property. ◀

If  $m < \ell^{c\gamma}/2$ , let  $S$  be the set promised by Claim 3. For every  $i \in [m]$ , the number of  $S$ -subfunctions of  $f_i$  is at most  $2^{2^{(1-\gamma/2) \log \ell}} = 2^{\ell^{1-\gamma/2}}$ , since each  $f_i$  reads at most  $(1 - \gamma/2) \log \ell$  variables from  $S$ . Each  $S$ -subfunction of  $f$  is uniquely determined by the  $S$ -subfunctions of  $f_1, \dots, f_m$ , and so  $f$  has at most  $2^{\ell^{1-\gamma/2} m}$   $S$ -subfunctions. By Claim 2, this means that  $m \geq \ell^{\gamma/2} > \ell^{c\gamma}/2$  – a contradiction. Hence  $C_{(1-\gamma)n}^2(f) \geq \ell^{c\gamma}/2 = \Omega(n^{c\gamma})$ , proving Theorem 2.

### 3.2.1 A matching upper bound for $f(x, y)$

We will now show that the lower bound from Theorem 2 is tight for the function  $f(x, y)$  defined above<sup>2</sup>, thus the exponential gap between  $C_k$  and  $C_k^2$  from Proposition 4 is inevitable.

► **Proposition 5.** *There exists  $c > 0$  such that for every  $0 < \gamma < 1/2$  and  $n$  sufficiently large,  $C_{(1-\gamma)n}^2 f(x, y) \leq n^{c\gamma}$  and  $C_{(1-\gamma)n} f(x, y) \leq c\gamma \log n$ .*

**Proof.** It is enough to prove the bound for  $C_{(1-\gamma)n}$  and invoke Proposition 4. We will outline the construction for  $\gamma = 1/2$  and then sketch how to adapt it to the general case. Divide the variables  $x$  into two equal subsets  $x_1$  and  $x_2$ . Let  $g_1$  be the function which, on inputs  $x_1$  and

<sup>2</sup> In the case when  $\gamma$  is fixed and  $n$  grows independently.

$y$ , outputs a  $\log \ell$ -bit string whose first bits equal  $x_1$  restricted to  $S_y$ . Define  $g_2$  similarly. This means that  $x_{S_y}$  can be recovered from  $x_2, y$  and the advice from  $g_1$ ; likewise for  $x_1, y$  and  $g_2$ . It is now easy to see that we can write  $f(x, y) = h_1(g_1, x_2, y) \vee h_2(g_2, x_1, y)$  with suitable  $h_1$  and  $h_2$ . This gives approximately  $\log n$  gates with fan-in approximately  $n/2$ .

In general, partition the variables  $x$  into  $r$  disjoint subsets  $a_1, \dots, a_r$  of nearly the same size. The gates will have access to the inputs  $y$  and  $x \setminus a_i$  for some  $i \in [r]$ . Note that for any  $\log \ell$  element subset  $S$  of  $x$ , there will exist two distinct  $a_i$  and  $a_j$  with  $|a_i \cap S|, |a_j \cap S| \leq 2 \log \ell / r$ . We can recover  $x_{S_y}$  from  $x \setminus a_i$  with an advice of  $2 \log \ell / r$  bits, and as above, compute  $f(x, y)$  using two gates depending  $y, x \setminus a_i$  and  $y, x \setminus a_j$  and  $2 \log n / r$  bits of advice each. The advice itself can be computed by gates which have access to either  $y, x \setminus a_1$  or  $y, x \setminus a_2$ . This gives a circuit with roughly  $8 \log n / r + r$  gates of fan-in  $(1 - 1/r)n$ ; this is at most  $10 \log n / r$  gates for fixed  $r$  and large enough  $n$ . ◀

### 3.3 Proof of Theorem 1

We will deduce a lower bound on  $C_k(f)$  from known NOF lower bounds. The main issue with the reduction to the NOF model is that any gate in the circuit may read an arbitrary set of inputs (perhaps even one bit from every party's forehead).

One way to simulate any circuit with linear fan-in and  $m$  gates using  $m$  parties is to associate every gate with a party and then greedily assign variables to parties, giving inputs of length  $\Omega(n/m)$  for each of the  $m$  parties. We manage to reduce the number of parties to  $O(m/\log n)$ , which enables us to obtain stronger lower bounds. This is done using the following Lemma:

▶ **Lemma 6.** *Let  $G \subseteq A \times B$  be a bipartite graph with  $|A| = m$ ,  $|B| = n$  and with every  $a \in A$  having degree at least  $\gamma n$ , where  $0 < \gamma < 1/100$  and  $n$  is sufficiently large with respect to  $\gamma^{-1}$ . If  $\log n \leq m \leq \log^2 n$ , then there exists  $p \leq 5m/\gamma$  and disjoint  $T_1, \dots, T_p \subseteq A$ ,  $S_1, \dots, S_p \subseteq B$ , each  $S_i$  of size at least  $n^{0.9}$ , such that  $A = \bigcup T_i$  and  $(T_i \times S_i) \subseteq G$  for every  $i \in [p]$ .*

**Proof.** We first prove the following:

▶ **Claim.** *If  $m \leq \log n$ ,  $G$  contains a complete bipartite graph with at least  $\gamma m/2$  vertices on the left and  $2n^{0.9}$  vertices on the right.*

**Proof.** Remove from  $B$  all vertices with degree  $\leq \gamma m/2$ . Since the graph has at least  $\gamma mn$  edges to begin with, the remaining set of vertices  $B'$  has size at least  $\gamma n/2$ . For  $M \subseteq A$ , let  $B(M)$  be the set of  $b \in B'$  such that  $b$  is connected to every  $a \in M$ . Hence,

$$B' = \bigcup_{|M|=\lceil \gamma m/2 \rceil} B(M).$$

Since  $m \leq \log n$  and  $\gamma < 1/100$ , the number of sets with  $|M| = \lceil \gamma m/2 \rceil$  is at most  $n^{0.09}$ . So there is such an  $M$  with  $|B(M)| \geq \frac{\gamma n/2}{n^{0.09}} \geq 2n^{0.9}$ , for  $n$  large enough. ◀

We iteratively apply the Claim to prove the Lemma. If  $m > \log n$ , choose an arbitrary  $\log n$ -element subset of  $A$  and let  $T_1 \times S_1$  be the complete graph guaranteed by the Claim. If  $m \leq \log n$ , apply the Claim directly to  $G$ . Remove from  $G$  all the vertices  $T_1$  and  $S_1$ , obtaining a new graph  $G_2 \subseteq A_2 \times B_2$ . Repeat this process  $p$  times to obtain graphs  $G_2, \dots, G_p$  until  $A_p = \emptyset$ . We claim that  $p \leq 5m/(\gamma \log n)$ . For such a small  $p$ , we have altogether removed  $o(n)$  vertices from  $B$  and so  $|B_i| \geq n(1 - o(1))$  for every  $i = 1, 2, \dots, p$ . Similarly, the degree of any  $a \in A_i$  is at least  $\gamma |B_i|/2$ . Hence, as long as  $|A_i| \geq \log |B_i|$ , we remove

at least  $\gamma \log n/4$  vertices from  $A_i$ . After at most  $4m/(\gamma \log n)$  steps, we then must have  $|A_i| < \log |B_i|$ . After this point,  $A_i$  decreases by at least the factor of  $(1 - \gamma/2)$ , and so the size drops below 1 in roughly  $\log \log n/\gamma$  steps, which is much smaller than  $m/\gamma$ . Finally, the size of every  $S_i$  is at least  $|B_i|^{0.9} = 2(n(1 - o(1)))^{0.9} \geq n^{0.9}$ , if  $n$  is large enough. ◀

Our hard function  $f(x, y)$  is defined as follows. It takes as inputs  $x \in \{0, 1\}^\ell$  and an auxiliary string  $y$ . We think of  $y$  as defining  $p \leq \log \ell$  disjoint subsets  $S_y^1, \dots, S_y^p$  of  $[\ell]$ , of equal size not exceeding  $\ell^{0.9}$ . Hence,  $y$  can be taken as roughly  $\ell^{0.9} \log^2 \ell$ -bit string. We define

$$f(x, y) := \text{GIP}(x_{S_y^1}, \dots, x_{S_y^p}).$$

$f(x, y)$  has  $n = \ell + O(\ell^{0.9} \log^2 \ell)$  variables. As before,  $x_{S_y^i}$  is the projection of  $x$  to  $S_y^i$ .

Suppose that for a fixed  $0 < \gamma$  and  $n$  sufficiently large,  $f(x, y)$  can be computed using  $m < \gamma \log^2 n/50$  non-input gates with fan-in  $n(1 - \gamma)$ . Take the graph  $G$  whose left vertices are the  $m$  gates of the circuit and the right vertices the  $\ell$  variables of  $x$ . There is an edge between a gate and a variable whenever the gate does *not* read the variable. Since  $y$  is much shorter than  $x$ , the degree of a gate in  $G$  is at least  $\gamma \ell/2$ . To apply the Lemma, we will assume  $\gamma < 1/100$  (otherwise the circuit is weaker) and that  $m \geq \log \ell$ . The Lemma shows that there exist disjoint sets of variables  $S_1, \dots, S_p$  with  $p \leq \log n/5$  and  $S_i = [\ell^{0.9}]$  such that each gate completely misses at least one set  $S_i$ . We can fix  $y$  so that  $y$  represents  $S_1, \dots, S_p$  and hence  $f(x, y)$  computes  $\text{GIP}(x_{S_1}, \dots, x_{S_p})$ . As observed in Section 2.1, the circuit gives an  $m$ -bit protocol for  $\text{GIP}(x_{S_1}, \dots, x_{S_p})$ . By the results of [3], this implies  $m \geq \Omega(\ell^{0.9} 2^{-2 \log \ell/5}) = \Omega(\sqrt{\ell})$ , contradicting the assumption that  $m < \gamma \log^2 n/50$ . This proves Theorem 1.

## 4 Connections to other models

Here we show how is our model connected to several disparate problems in complexity theory.

### 4.1 Circuits of linear size and logarithmic depth

Obviously,  $C_k(f) \leq C_2(f)$ , so any super-linear lower bound in our model would give a super-linear lower bound for circuits of fan-in 2. However, even a linear lower bound on our model would give a function that cannot be computed by a linear sized logarithmic depth circuit:

► **Proposition 7.** *If  $f$  has a fan-in 2 circuit of linear size and logarithmic depth, then for any  $\epsilon > 0$ ,  $C_{n^\epsilon}(f) < O\left(\frac{n \log(1/\epsilon)}{\log \log n}\right)$ .*

Valiant [21] showed that any (fan-in 2) circuit of linear size and logarithmic depth contains a set  $T$  of  $O\left(\frac{n \log(1/\epsilon)}{\log \log n}\right)$  gates such that every path of length  $\epsilon \log n$  in the circuit must touch a gate from the set. Since every such gate in  $T$  can be computed from at most  $n^\epsilon$  other gates from  $T$  and the inputs, we obtain Proposition 7.

### 4.2 Oblivious branching programs

An oblivious branching program of width  $w$  and length  $\ell$  is a directed graph with vertices partitioned into  $\ell$  layers  $L_1, \dots, L_\ell$ . Each layer is associated with an input variable. Every vertex in  $L_i$  has out-degree 2, with the edges labeled 0, 1. Every vertex of  $L_\ell$  is labeled with

an output value. The program is executed by starting at the first vertex of  $L_1$ , and reading the variables in turn to find a path through the program until the output is determined.

Barrington [4] showed that every logarithmic depth circuit (of fan-in 2) can be simulated by a branching program with  $w = 5, \ell = \text{poly}(n)$ . Thus it is very interesting to prove super-polynomial lower bounds on such programs. A line of work has proved *time-space tradeoffs* on such programs. Alon and Maass [2] used reductions to Ramsey theory to show that any program for computing the majority function must have  $\ell \log w \geq \omega(n \log n)$ . Babai, Nisan and Szegedy [3] proved a lower bound of  $\ell \log w \geq \Omega(n \log^2 n)$  by reductions to the Number-on-Forehead communication model. Beame and Vee [6] simplified the proof of this last bound. No better lower bound on  $\ell \log w$  is known, to our knowledge.

Our results give lower bounds that match those of [3] via the following proposition:

► **Proposition 8.** *If  $f$  can be computed by an oblivious branching program of width  $w < 2^{\epsilon n/2}$  and length  $\ell$ , then  $C_{\epsilon n}(f) \leq \frac{2\ell \log w}{\epsilon n}$ .*

The first  $\log w$  gates of the circuit read the first  $\epsilon n/2$  variables read by the program and together compute the name of the vertex reached after those layers. The next  $\log w$  gates read the outputs of the previous gates and the next  $\epsilon n/2$  variables, to compute the name of the vertex in layer  $L_{\epsilon n}$ . Continue in this way until all of the program has been simulated. Thus we obtain a lower bound of  $\ell \log w \geq \Omega(n \log^2 n)$  on the length of the program using Proposition 8 and Theorem 1. Any lower bound of the type  $C_{\epsilon n}(f) = \omega(\log^2 n)$  would give new time-space tradeoffs for branching programs.

### 4.3 $AC_0$

An  $AC_0$  circuit is a circuit of constant depth that uses AND, OR-gates of unbounded fan-in and NOT-gates. As negations can be moved to the leaves, the depth of  $AC_0$  circuit is defined as the largest number of AND, OR-gates on a path in the circuit. Any size  $s$   $AC_0$  circuit can be simulated by a size  $s^2$  circuit with gates of fan-in 2.

Beautiful methods have been developed to prove lower bounds on these circuits [13, 19, 20]. The best known lower bounds for a depth  $d$  circuit are of the type  $2^{\Omega(n^{1/(d-1)})}$ . The following proposition shows that a truly exponential lower bound would give a linear lower bound in our model.

► **Proposition 9.** *For every  $f$  and  $k$ ,  $f$  can be computed by a depth-3  $AC_0$  circuit of size  $O(kC_k(f) \cdot 2^{C_k(f)+k})$ .*

To see this, observe that the function  $g$  defined in the proof of Proposition 4 can be computed by a monotone formula in disjunctive normal form, with  $C_k(f) \cdot 2^{C_k(f)}$  leaves. Furthermore, each  $f_{i,\sigma}$  depends on  $k$  variables, and hence it can be computed by a formula in conjunctive normal form, with  $k \cdot 2^k$  leaves. This gives depth-3  $AC_0$  formula with  $kC_k(f) \cdot 2^{C_k(f)+k}$  leaves.

Proposition 9 implies that if  $f$  cannot be computed by a depth-3  $AC_0$  circuit of size  $2^{2k}$ , then  $C_k(f) \geq \Omega(k)$ . Hence, a lower bound of  $2^{\omega(\sqrt{n})}$  for depth-3  $AC_0$  circuits would give a non-trivial lower bound on  $C_{\sqrt{n}}(f)$ , and a lower bound of  $2^{\Omega(n)}$  would yield a linear lower bound on  $C_{\epsilon n}(f)$ . In addition, the latter  $f$  cannot have a linear sized circuit of logarithmic depth by Proposition 7; an observation already made by Valiant [21].

### 4.4 Extractors for varieties

Given a field  $\mathbb{F}$ , a variety is a set of the form  $\{x \in \mathbb{F}^n : f_1(x) = f_2(x) = \dots = f_m(x) = 0\}$ , where  $f_1, \dots, f_m$  are polynomials. For a finite field  $\mathbb{F}$ , an *extractor for varieties* is a function



$f : \mathbb{F}^n \rightarrow \{0, 1\}$  which is non-constant on any sufficiently large variety defined by low-degree polynomials.

Dvir [11] showed how to use bounds on exponential sum estimates by Deligne [10] to obtain extractors for varieties. Working over a prime field of size  $p$ , he shows that if  $\rho > 1/2$  is a constant, and  $V \subseteq \mathbb{F}^n$  is a variety of size  $p^{\rho n}$  defined by polynomials of degree  $\rho n$ , then there is an efficiently computable extractor for such varieties, as long as  $p$  is polynomially large in  $n$ . Here we show that such a result for  $p = 2$  would imply non-trivial circuit lower bounds.

► **Proposition 10.** *Let  $p = 2$ . If  $f$  is an extractor for varieties of size  $2^{\rho n}$  defined by degree  $k$  polynomials, then  $C_k(f) > (1 - \rho)n$ .*

**Proof.** Suppose there is a circuit computing  $f$  with  $m$  gates of fan-in  $k$ . By averaging, there must exist some evaluation of the gates which is consistent with  $2^{n-m}$  input strings. We now define a variety using  $m$  polynomials as follows. Each polynomial checks that the input is consistent with the evaluations of a single gate. Since every such polynomial depends on at most  $k$  variables, and it can be taken multilinear, it has degree at most  $k$ . Thus we obtain a variety of size  $2^{n-m}$  defined by degree  $k$  polynomials on which  $f$  is constant. So it must be that  $n - m < \rho n \Rightarrow m > (1 - \rho)n$ . ◀

By Proposition 7, any such extractor cannot be computed by linear sized logarithmic depth circuits of fan-in 2.

**Acknowledgements.** We thank Paul Beame, Parikshit Gopalan, Makrand Sinha and Avi Wigderson for useful comments.

---

## References

- 1 Miklós Ajtai. Determinism versus nondeterminism for linear time RAMs with memory restrictions. *Journal of Computer and System Sciences*, 65(1):2–37, 2002.
- 2 Noga Alon and Wolfgang Maass. Meanders, ramsey theory and lower bounds for branching programs. In *Proceedings of the 27th Annual Symposium on Foundations of Computer Science, FOCS*, pages 410–417, 1986.
- 3 László Babai, Noam Nisan, and Mario Szegedy. Multiparty protocols, pseudorandom generators for logspace, and time-space trade-offs. *J. Comput. Syst. Sci.*, 45(2):204–232, 1992.
- 4 David A. Mix Barrington. Bounded-width polynomial-size branching programs recognize exactly those languages in  $\text{NC}^1$ . *J. Comput. Syst. Sci.*, 38(1):150–164, 1989.
- 5 Paul Beame, Paraschos Koutris, and Dan Suciu. Communication steps for parallel query processing. In *Proceedings of the 32nd ACM Symposium on Principles of Database Systems, PODS*, pages 273–284, 2013.
- 6 Paul Beame and Erik Vee. Time-space tradeoffs, multiparty communication complexity, and nearest-neighbor problems. In *Proceedings on 34th Annual ACM Symposium on Theory of Computing, STOC*, pages 688–697, 2002.
- 7 Allan Borodin, Alexander A. Razborov, and Roman Smolensky. On lower bounds for read- $K$ -times branching programs. *Computational Complexity*, 3:1–18, 1993.
- 8 A. Chandra, M. Furst, and R. Lipton. Multi-party protocols. In *Proceedings of the 15th Annual ACM Symposium on Theory of computing, STOC*, pages 94–99, 1983.
- 9 D. Y. Cherukhin. The lower estimate of complexity in the class of schemes of depth 2 without restrictions on a basis. *Moscow University Mathematics Bulletin*, 60(4):42–44, 2005.
- 10 Pierre Deligne. La conjecture de Weil : I, 1974.



- 11 Zeev Dvir. Extractors for varieties. *Computational Complexity*, 21(4):515–572, 2012.
- 12 Oded Goldreich and Avi Wigderson. On the size of depth-three boolean circuits for computing multilinear functions. *Electronic Colloquium on Computational Complexity, ECCC*, 2013.
- 13 Johan Hastad. Almost optimal lower bounds for small depth circuits. In *Proceedings of the 18th Annual ACM Symposium on Theory of Computing, STOC*, pages 6–20, 1986.
- 14 S. Jukna. *Extremal combinatorics, with applications in computer science*. Springer-Verlag, 2001.
- 15 E. I. Nechiporuk. A boolean function. *Sov.Math.Dokl.*, 7(4):999–1000, 1966.
- 16 Elizaveta Okolnishnikova. On lower bounds for branching programs. *Siberian Advances in Mathematics*, 3(1):152–166, 1993.
- 17 P. Pudlák, V. Rödl, and J. Sgall. Boolean circuits, tensor ranks, and communication complexity. *SIAM J. Comput.*, 26(3):605–633, 1997.
- 18 J. Radhakrishnan and A. Ta-Shma. Bounds for dispersers, extractors, and depth-two superconcentrators. *SIAM J. Discrete Math.* 13(1): 2–24, 13(1):2–24, 200.
- 19 Alexander Razborov. Lower bounds on the size of bounded depth circuits over a complete basis with logical addition. *MATHNASUSSR: Mathematical Notes of the Academy of Sciences of the USSR*, 41, 1987.
- 20 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*, pages 77–82, 1987.
- 21 Leslie G. Valiant. Graph-theoretic arguments in low-level complexity. In *Mathematical Foundations of Computer Science, MFCS*, volume 53 of *Lecture Notes in Computer Science*, pages 162–176, 1977.

# Correlation Bounds Against Monotone NC<sup>1</sup>

Benjamin Rossman\*

National Institute of Informatics, Tokyo, Japan

Simons Institute for the Theory of Computation, Berkeley, USA

rossman@nii.ac.jp

---

## Abstract

---

This paper gives the first correlation bounds under product distributions, including the uniform distribution, against the class mNC<sup>1</sup> of polynomial-size  $O(\log n)$ -depth monotone circuits. Our main theorem, proved using the pathset complexity framework introduced in [56], shows that the average-case  $k$ -CYCLE problem (on Erdős-Rényi random graphs with an appropriate edge density) is  $\frac{1}{2} + \frac{1}{\text{poly}(n)}$  hard for mNC<sup>1</sup>. Combining this result with O’Donnell’s hardness amplification theorem [43], we obtain an explicit monotone function of  $n$  variables (in the class mSAC<sup>1</sup>) which is  $\frac{1}{2} + n^{-1/2+\varepsilon}$  hard for mNC<sup>1</sup> under the uniform distribution for any desired constant  $\varepsilon > 0$ . This bound is nearly best possible, since every monotone function has agreement  $\frac{1}{2} + \Omega(\frac{\log n}{\sqrt{n}})$  with some function in mNC<sup>1</sup> [44].

Our correlation bounds against mNC<sup>1</sup> extend smoothly to non-monotone NC<sup>1</sup> circuits with a bounded number of negation gates. Using Holley’s monotone coupling theorem [30], we prove the following lemma: with respect to any product distribution, if a balanced monotone function  $f$  is  $\frac{1}{2} + \delta$  hard for monotone circuits of a given size and depth, then  $f$  is  $\frac{1}{2} + (2^{t+1} - 1)\delta$  hard for (non-monotone) circuits of the same size and depth with at most  $t$  negation gates. We thus achieve a lower bound against NC<sup>1</sup> circuits with  $(\frac{1}{2} - \varepsilon)\log n$  negation gates, improving the previous record of  $\frac{1}{6}\log\log n$  [7]. Our bound on negations is “half” optimal, since  $\lceil\log(n+1)\rceil$  negation gates are known to be fully powerful for NC<sup>1</sup> [3, 21].

**1998 ACM Subject Classification** F.1.3 Complexity Measures and Classes

**Keywords and phrases** circuit complexity, average-case complexity

**Digital Object Identifier** 10.4230/LIPIcs.CCC.2015.392

## 1 Introduction

The majority of research in boolean circuit complexity is focused on restricted classes of circuits. Super-polynomial lower bounds are known in two basic settings: *bounded-depth circuits* (i.e. AC<sup>0</sup>) [1, 24] and *monotone circuits* [51]. For another natural class, *deMorgan formulas* (tree-like circuits with fan-out 1), nearly cubic  $n^{3-o(1)}$  lower bounds are known [28]. For bounded-depth circuits as well as deMorgan formulas, the state-of-the-art worst-case lower bounds (obtained in the 1980’s and 90’s) have recently been matched by tight *average-case lower bounds*, also known as *correlation bounds*, under the uniform distribution. It is now known that

- PARITY is  $\frac{1}{2} + 2^{-\Omega(n/(\log S)^{d-1})}$  hard for depth- $d$  (unbounded fan-in) circuits of size  $S$  [29],
- there is an explicit function in P which is  $\frac{1}{2} + 2^{-\Omega(r)}$  hard for deMorgan formulas of size  $n^{3-o(1)}/r^2$  [40].

---

\* Supported by the JST ERATO Kawarabayashi Large Graph Project and the Simons Institute Research Fellowship.



(A boolean function  $f$  is said to be  $\gamma$ -hard for a class of circuits  $\mathcal{C}$  under a distribution  $\mu$  if  $\mathbb{P}_{x \sim \mu}[f(x) = \mathfrak{C}(x)] \leq \gamma$  for every circuit  $\mathfrak{C} \in \mathcal{C}$ . By default  $\mu$  is the uniform distribution and  $\gamma$  is typically expressed as  $\frac{1}{2} + \delta$  or  $1 - \delta$  where  $\delta(n) \rightarrow 0$ .)

In the monotone setting, there is an excellent knowledge of worst-case lower bounds: a long line of works [4, 8, 27, 37, 45, 48, 46, 50, 51] (among many others) have achieved separations between the monotone versions of nearly all the important complexity classes, as defined by Grigni and Sipser [26]. However, when it comes to average-case lower bounds under the uniform distribution or any product distribution, essentially nothing has been known; it is still open, for instance, whether any monotone function in NP is  $1 - \frac{1}{\text{poly}(n)}$  hard for polynomial-size monotone circuits. This represents a major gap in the basic understanding of monotone computation, given the importance of product distributions in the monotone setting. Product distributions are believed to be a natural source of hard instances for many monotone problem:  $k$ -SAT and  $k$ -CLIQUE are famously conjectured to be hard-on-average at an appropriate threshold density [38]. Product distributions are also significant in the real analysis of monotone functions (see the FKG inequality [22], the Bollobás-Thomason theorem [18], Friedgut’s threshold theorem [23], etc.)

## 1.1 Previous Work

Many of the aforementioned *worst-case* lower bounds in the monotone setting can be viewed as average-case lower bounds under specific *non-product* distributions. To take one example, consider Razborov’s celebrated lower bound for the  $k$ -CLIQUE function [51]. Let  $\mu$  denote the distribution on  $n$ -vertex graphs which, half of the time, is a uniform random  $k$ -clique, and the other half is a uniform random  $(k - 1)$ -coclique (i.e. complete  $(k - 1)$ -partite graph). The result of [51] (together with the quantitative improvement [4]) shows that, for all  $3 \leq k \leq n^{1/4}$ ,  $k$ -CLIQUE is  $\frac{1}{2} + n^{-\Omega(\sqrt{k})}$  hard under  $\mu$  for the class mP of polynomial-size monotone circuits (for  $k \leq \log n$ , this bound improves to  $\frac{1}{2} + n^{-\Omega(k)}$ ). (Correlation bounds under similar (non-product) distributions were recently obtained for monotone classes within mP [20, 25], strengthening previous worst-case separations among these classes.)

Toward the goal of worst-case lower bounds, this distribution  $\mu$  has a very sensible property: it is supported entirely on *minterms* (minimal 1-instances, i.e.,  $k$ -cliques) and *maxterms* (maximal 0-instances, of which  $(k - 1)$ -cocliques are a subset). Thus  $\mu$  exploits monotonicity in the strongest possible way. However, there is something backwards about  $\mu$ : every 1-instance has Hamming weight  $\binom{k}{2}$  ( $\leq \sqrt{n}$ ), which is less the minimum Hamming weight  $\binom{k-1}{2} \binom{n}{k-1}^2$  ( $\geq n^2/2$ ) of any 0-instance. It follows that  $k$ -CLIQUE is equivalent under  $\mu$  to the *anti-monotone* threshold function  $\text{THR}_{<n^2/2}$  (which is 1 on graphs with fewer than  $n^2/2$  edges). Therefore, even though  $k$ -CLIQUE is hard under  $\mu$  for monotone circuits, it is easy under  $\mu$  for polynomial-size circuits with a single negation gate (in fact,  $\text{THR}_{<n^2/2}$  is computable by polynomial-size formulas with a single negation [3]).

This discomfort was addressed in work of Amano and Maruoka [7], who extended the  $k$ -CLIQUE lower bound of [4, 51] to polynomial-size circuits with  $\frac{1}{6} \log \log n$  negation gates by considering a modified distribution  $\mu'$  (a certain convex combination, over various  $\ell \in \{k, \dots, n\}$ , of  $\ell$ -cliques and  $(k - 1)$ -cocliques supported on sets of size  $\ell$ ). While the core of the proof in [7] is still a monotone circuit lower bound for cliques vs. cocliques, this result contributed an insight that sufficiently strong lower bounds against monotone circuits imply lower bounds against negation-limited boolean circuits (we capitalize on this insight in Lemma 1.3).

A more natural distribution for the average-case analysis of  $k$ -CLIQUE is given by the Erdős-Rényi random graph  $G(n, p)$ . Here we take  $p$  ( $= p(n, k)$ ) to be the unique “ $\frac{1}{2}$ -threshold”

such that  $\mathbb{P}[G(n, p) \text{ contains a } k\text{-clique}] = \frac{1}{2}$ . (Note that  $G(n, p)$  is a product distribution on  $\{0, 1\}^{\binom{n}{2}}$ .) Karp [38] famously conjectured that  $k$ -CLIQUE is hard-on-average under  $G(n, p)$  (in the regime  $k \approx 2 \log n$ ). Previous work of the author [54] confirmed this conjecture in the (non-monotone) AC<sup>0</sup> setting by showing that  $k$ -CLIQUE is  $\frac{1}{2} + n^{-\Omega(k)}$  hard under  $G(n, p)$  for AC<sup>0</sup> (polynomial-size constant-depth circuits) for all  $k \leq \log^{1/2} n$ . Follow-up work of the author [55] combined the technique of [54] with the “approximation method” framework of Razborov [51] to prove a correlation bound against monotone circuits under a different distribution  $\nu$  on  $n$ -vertex graphs: half of the time,  $\nu$  is  $G(n, p)$  plus a uniform random planted  $k$ -clique, and the other half  $\nu$  is  $G(n, 2p)$  conditioned on being  $k$ -clique-free. The result of [55] shows that  $k$ -CLIQUE is  $\frac{1}{2} + n^{-\Omega(k)}$  hard under  $\nu$  for mP for all  $k \leq \log^{1/2} n$ . While  $\nu$  is (morally speaking) similar to  $G(n, p)$ , it is unfortunately not a product distribution and suffers the same shortcoming as  $\mu$ : the 0-instances and 1-instances under  $\nu$ , although no longer minterms and maxterms, are nevertheless separable with high probability by an anti-monotone threshold function (in this case  $\text{THR}_{< \frac{3}{2} \binom{n}{2} p}$ ). It was left as an open problem in [55] to prove a correlation bound against mP for  $k$ -CLIQUE under  $G(n, p)$ . In the present paper, we do not succeed in this task; however, we prove a correlation bound against a significant subclass of mP (mNC<sup>1</sup>) for a different monotone graph property ( $k$ -CYCLE) under an appropriate product distribution.

### 1.2 Our Results

Our main theorem is a correlation bound for the average-case  $k$ -CYCLE problem against the class mNC<sup>1</sup> of polynomial-size  $O(\log n)$ -depth monotone circuits (equivalently, polynomial-size monotone formulas). Rather than the standard Erdős-Rényi random graph, we find it convenient to restrict attention to “ $C_k$ -partite” input graphs with  $kn$  vertices and  $kn^2$  potential edges (Def. 4.2). For the average-case analysis of  $k$ -CYCLE, we consider a random  $C_k$ -partite graph, denoted  $\Gamma$ , in which each potential edge is independently included with probability  $p$  where  $p$  is the unique “ $\frac{1}{2}$ -threshold” such that  $\mathbb{P}[\Gamma \text{ contains a } k\text{-cycle}] = \frac{1}{2}$ . (Note:  $p \sim c_k/n$  for a constant  $c_k$  depending on  $k$ .) A monotone function  $f$  on  $kn^2$  variables is said to compute  $k$ -CYCLE on  $\Gamma$  with *advantage*  $\delta$  if  $\mathbb{P}[f(\Gamma) = k\text{-CYCLE}(\Gamma)] \geq \frac{1}{2} + \delta$ .

► **Theorem 1.1 (Main Theorem).** *For all  $k \leq \log \log n$ , if a monotone formula computes  $k$ -CYCLE on  $\Gamma$  with advantage  $n^{-1/2+c}$ , then it has size  $n^{\Omega(c \log k)}$ . In particular,  $\log \log n$ -CYCLE is  $\frac{1}{2} + n^{-1/2+o(1)}$  hard under  $\Gamma$  for monotone formulas of size  $n^{o(\log \log \log n)}$  (and hence for mNC<sup>1</sup>).*

The lower bound in Theorem 1.1 is essentially tight, since  $k$ -CYCLE is computable (in the worst-case) by monotone formulas of size  $n^{O(\log k)}$ , as well as by polynomial-size  $O(\log k)$ -depth monotone circuits with semi-unbounded fan-in (i.e. binary AND gates and unbounded OR gates). This places  $k$ -CYCLE in the class mSAC<sup>1</sup>. (In terms of space complexity,  $k$ -CYCLE is computable in NL as well as Ave-L, as defined in [12].) Theorem 1.1 thus gives a very sharp average-case separation of mNC<sup>1</sup> from higher complexity classes.

As a corollary of Theorem 1.1, we obtain nearly optimal correlation bounds against mNC<sup>1</sup> under the *uniform distribution*. Note that the random graph  $\Gamma$ , while a product distribution, is not the uniform distribution on  $\{0, 1\}^{kn^2}$ ; moreover, the correlation bound in Theorem 1.1 is only  $\frac{1}{2} + (kn^2)^{-1/4+o(1)}$  in terms of the input size  $kn^2$ . Nevertheless, using O’Donnell’s hardness amplification theorem [43], we have the following result:

► **Corollary 1.2.** *For every  $\varepsilon > 0$ , there is an explicit monotone function of  $N$  variables (in the class mSAC<sup>1</sup>) which is  $\frac{1}{2} + N^{-1/2+\varepsilon}$  hard for mNC<sup>1</sup> under the uniform distribution.*

This function is the composite function<sup>1</sup>  $\text{TRIBES} \otimes \log \log n\text{-CYCLE} \otimes p\text{-BIAS}$  on  $N = \text{poly}(n)$  variables where

- $p\text{-BIAS} : \{0, 1\}^n \rightarrow \{0, 1\}$  is any  $n$ -term monotone DNF with exactly  $\lceil p2^n \rceil$  satisfying assignments,<sup>2</sup>
- $\text{TRIBES} : \{0, 1\}^{n^c} \rightarrow \{0, 1\}$  is the “tribes” function of Ben-Or and Linial [13] on  $n^c$  variables, where  $c (= \Omega(1/\varepsilon))$  is a sufficiently large constant.

See O’Donnell’s paper [43] for background on the hardness amplification theorem which yields Corollary 1.2 from Theorem 1.1. We only remark that all results in [43], while stated in terms of the class NP, apply equally to  $\text{mNC}^1$ . This observation relies on the fact that  $\text{MAJ} \in \text{mNC}^1$  [3, 63] (where MAJ is the majority function); this is essential for the application of Implagliazzo’s “hard-core set” theorem [31, 39], which is a main tool in [43].

The correlation bound in Corollary 1.2 is nearly best possible under the uniform distribution, as O’Donnell and Wimmer [44] have shown that every monotone function  $\{0, 1\}^n \rightarrow \{0, 1\}$  has agreement  $\frac{1}{2} + \Omega(\frac{\log n}{\sqrt{n}})$  with one of functions  $0, 1, x_1, \dots, x_n, \text{MAJ}$ . Since these functions are all in  $\text{mNC}^1$ , it follows that no monotone function is  $\frac{1}{2} + o(\frac{\log n}{\sqrt{n}})$  hard for  $\text{mNC}^1$ . Corollary 1.2 shows that this correlation bound is nearly achieved by an explicit monotone function. (By counting arguments, there exist (non-explicit) monotone functions achieving similar correlation bounds [9, 36].)

Finally, we extend these results to non-monotone circuits with a bounded number of negation gates, by means of a general lemma on correlation bounds under product distribution. In fact, our observation applies to the broader class of distributions  $\mu$  on  $\{0, 1\}^n$  which satisfy the *FKG lattice condition* [22] if

$$\mu(x)\mu(y) \leq \mu(x \wedge y)\mu(x \vee y) \quad \text{for all } x, y \in \{0, 1\}^n. \quad (1)$$

Note that every product distribution satisfies (1) with equality.

► **Lemma 1.3.** *Let  $\mu$  be a distribution which satisfies the FKG lattice condition (1) and let  $f$  be a monotone function which is balanced under  $\mu$  (i.e.  $\mathbb{E}_\mu(f) = \frac{1}{2}$ ). If  $f$  is  $\frac{1}{2} + \delta$  hard under  $\mu$  for monotone circuits of a given size and depth, then  $f$  is  $\frac{1}{2} + (2^{t+1} - 1)\delta$  hard under  $\mu$ , up to the same size and depth, for (non-monotone) circuits with  $t$  negation gates.*

Via Lemma 1.3, the correlation bound of Corollary 1.2 extends to  $\text{NC}^1$  circuits with  $(\frac{1}{2} - \varepsilon) \log n$  negation gates.

► **Corollary 1.4.** *For every  $\varepsilon > 0$ , there is an explicit function in  $\text{mSAC}^1$  which is  $\frac{1}{2} + o(1)$  hard for  $\text{NC}^1$  circuits with  $(\frac{1}{2} - \varepsilon) \log n$  negations under the uniform distribution.*

Corollary 1.4 is half optimal, in the sense that  $\text{NC}^1$  circuits with  $\lceil \log(n+1) \rceil$  negations are known to be equivalent to full  $\text{NC}^1$  by well-known results of Markov [42] and Fischer [21] (again using the fact that  $\text{MAJ} \in \text{NC}^1$ ). This improves a previous  $\frac{1}{6} \log \log n$  lower bound of Amano and Maruoka [7] on the negation-limited complexity of an explicit monotone function  $\{0, 1\}^n \rightarrow \{0, 1\}$  (however, unlike Corollary 1.4, the result of [7] applies to polynomial-size circuits of unbounded depth). For multi-output monotone functions  $\{0, 1\}^n \rightarrow \{0, 1\}^n$ , Jukna

<sup>1</sup> For boolean functions  $h : \{0, 1\}^l \rightarrow \{0, 1\}$  and  $g : \{0, 1\}^m \rightarrow \{0, 1\}$ , the composite function  $g \otimes h : (\{0, 1\}^l)^m \rightarrow \{0, 1\}$  is defined by  $(g \otimes h)(y_1, \dots, y_m) = g(h(y_1), \dots, h(y_m))$ .

<sup>2</sup> It is an easy exercise to show that there is an  $n$ -term monotone DNF with exactly  $m$  satisfying assignments for every  $m \in \{0, \dots, 2^n\}$ . Note that  $p\text{-BIAS}$  generates a single  $p'$ -biased bit from the uniform distribution on  $\{0, 1\}^n$  where  $p \leq p' < p + 2^{-n}$ .

[34] proved a worst-case lower bound of  $\log n - O(\log \log n)$ . (There is an extensive literature on negation-limited complexity; see Chapter 10 of [35] and papers [11, 14, 16, 36, 64, 67] besides those already mentioned.)

### 1.3 Overview

We present an outline of the paper, highlighting the main ideas in the proof of Theorem 1.1.

#### Persistent Minterms

In Section 3 we introduce the key notion of *persistent minterms* of a monotone function  $f$  under an increasing sequence of monotone restrictions. Formally, we consider the sequence of monotone functions  $f^{\vee \rho_0} \leq f^{\vee \rho_1} \leq \dots \leq f^{\vee \rho_m}$  where  $\rho_0 \leq \rho_1 \leq \dots \leq \rho_m$  are elements in  $\{0, 1\}^n$  and  $f^{\vee \rho_i}(x) := f(x \vee \rho_i)$ . An element  $x \in \{0, 1\}^n$  of Hamming weight  $|x| = k$  is called a  $d$ -*persistent minterm* of  $f$  under  $\vec{\rho}$  if it is a common minterm of  $\binom{d+k-1}{k-1}$  many functions  $f^{\vee \rho_i}$ .

Persistent minterms behave like ordinary minterms under operations  $\vee$  and  $\wedge$  (Lemma 3.4). In addition, persistent minterms have the desirable property of being “noise-insensitive” in a certain sense. Suppose  $\xi^{(1)}, \dots, \xi^{(m)}$  are independent samples from a distribution of random “noise” over  $\{0, 1\}^n$ . If we now define  $\rho_i$  by  $\xi^{(1)} \cup \dots \cup \xi^{(i)}$ , then every persistent minterm is noise-insensitive, insofar as it has survived at least one hit of random noise. This translates into a lemma to the effect that every monotone function whatsoever has “few” persistent minterms with high probability (Lemma 5.13).

#### Average-Case $k$ -CYCLE

In Section 4 we consider the average-case  $k$ -CYCLE problem on the random graph  $\Gamma$  (i.e. the  $p$ -biased product distribution on  $\{0, 1\}^{kn^2}$  for appropriate  $p \sim 1/n$ ). We introduce an auxiliary random graph  $\Xi_\ell$  consisting of  $\ell$  ( $\ll \sqrt{n}$ ) independent paths of length  $k - 1$ . Crucially,  $\Xi_\ell$  lives “within the variance” of the random graph  $\Gamma$ , in the sense that  $\Gamma$  and  $\Gamma \cup \Xi_\ell$  have small total variation distance. Roughly speaking, we are able to show: if a monotone function  $f$  has correlation  $\gg \ell k^2 / \sqrt{n}$  with  $k$ -CYCLE under  $\Gamma$ , then a non-negligible fraction (at least  $1/\sqrt{n}$ ) of  $k$ -cycles are persistent minterms of  $f$  with respect to random noise  $\Xi_\ell$  (Lemma 4.5).

#### Pathset Complexity

In Section 5 we present the pathset complexity framework and state a lower bound proved in [56]. Very roughly speaking, for a subgraph  $A = (V_A, E_A)$  of the  $k$ -cycle, a *pathset over  $A$*  is a set of isomorphic copies of  $A$  embedded (as “sections”) in  $V_A \times [n]$ . *Pathset complexity* is a (formula-like) complexity measure on pathsets with respect to operations  $\cup$  and  $\bowtie$  (union and join). Crucially, pathsets are subject to a collection of density constraints called *smallness*; this bottleneck accounts for the high complexity of constructing pathsets beyond a certain size.

The pathset complexity framework was introduced in [56] for the purpose of separating formula-size and circuit-size within AC<sup>0</sup>. The technique is tailored to the formula complexity of the (virtually equivalent) average-case  $k$ -STCONN /  $k$ -CYCLE problems. The paper [56] proves a lower bound of  $n^{\Omega(\log k)}$  on the pathset complexity of any sufficiently dense pathset over the  $k$ -path /  $k$ -cycle (Theorem 5.8).

Our correlation bound against  $\text{mNC}^1$  (Theorem 1.1) is proved by reduction to this pathset complexity lower bound. Given a monotone formula which correlates well with  $k$ -CYCLE under  $\Gamma$ , we define (random) pathsets at all gates of the formula in terms of persistent minterms. We show (Lemma 5.13) that all of these pathsets satisfy the smallness constraint with high probability. In this way, we are able to obtain a formula-size lower bound from pathset complexity. The proof of Theorem 1.1 is given in Section 6; proofs of various lemmas are included in appendices (due to space limitation, two appendices which appear in the full version of this paper have been removed from this version).

## 2 Preliminaries

Let  $\mathbb{N} = \{0, 1, 2, \dots\}$ . For  $n \in \mathbb{N}$ , let  $[n] = \{1, \dots, n\}$ . We write  $\ln(\cdot)$  for the natural logarithm and  $\log(\cdot)$  for the base-2 logarithm. For random variables  $X$  and  $Y$ , notation  $X \stackrel{d}{=} Y$  expresses “ $X$  and  $Y$  are identically distributed”.

► **Definition 2.1** (Monotone Functions, Minterms, Monotone Restrictions).  $\mathbb{B}_n^+$  denotes the lattice of monotone (non-decreasing) boolean functions  $\{0, 1\}^n \rightarrow \{0, 1\}$ .  $f, g$  represent functions in  $\mathbb{B}_n^+$ .  $f \leq g$  denotes  $f(x) \leq g(x)$  for all  $x \in \{0, 1\}^n$ . For  $f \in \mathbb{B}_n^+$  and  $x \in \{0, 1\}^n$ , we say that  $x$  is a *minterm* of  $f$  if  $f(x) = 1$  and  $f(x') = 0$  for all  $x' < x$ . The set of minterms of  $f$  is denoted by  $\mathcal{M}(f)$ . (Note that  $\mathcal{M}(\cdot)$  gives a bijection from  $\mathbb{B}_n^+$  to anti-chains in  $\{0, 1\}^n$ .) For  $f \in \mathbb{B}_n^+$  and  $\rho \in \{0, 1\}^n$ , we denote by  $f^{\vee\rho}$  be the monotone function  $f^{\vee\rho}(x) := f(x \vee \rho)$ . (Note that  $f \leq f^{\vee\rho}$ .) In this context, we view  $\rho \in \{0, 1\}^n$  as a “monotone restriction” which sets some variables to 1 (namely,  $i \in [n]$  such that  $\rho_i = 1$ ) and leaves the remaining variables unset.

► **Lemma 2.2** (Minterm Lemma). For all  $f, g \in \mathbb{B}_n^+$ ,

$$\mathcal{M}(f \vee g) \subseteq \mathcal{M}(f) \cup \mathcal{M}(g), \quad \mathcal{M}(f \wedge g) \subseteq \{x \vee y : x \in \mathcal{M}(f), y \in \mathcal{M}(g)\}. \quad (2)$$

In other words, every minterm of  $f \vee g$  is a minterm of  $f$  or a minterm of  $g$ , and every minterm of  $f \wedge g$  is the disjunction of a minterm of  $f$  and a minterm of  $g$ . (This is easy to see, for instance, by thinking of the DNF representations of  $f$  and  $g$ .)

► **Definition 2.3** (Monotone Formulas). A *monotone formula* on  $n$  variables is a finite rooted binary tree whose leaves (inputs) are labeled by elements of  $[n] \cup \{0, 1\}$  and whose non-leaves (gates) are labeled  $\wedge$  or  $\vee$ . (In this paper all AND and OR gates have fan-in 2.) Every monotone formula  $\Phi$  on  $n$  variables computes a monotone function in  $\mathbb{B}_n^+$  (in the usual way). For  $x \in \{0, 1\}^n$ , we write  $\Phi(x)$  for the value of the monotone function computed by  $\Phi$  on input  $x$ .  $\text{Sub}(\Phi)$  denotes the set of (syntactic) sub-formulas of  $\Phi$ . For example, if  $\Phi$  is the formula  $\Psi \wedge \Psi$ , then  $\text{Sub}(\Phi)$  contains both (left and right) copies of  $\Psi$ .  $\text{Leaves}(\Phi) (\subseteq \text{Sub}(\Phi))$  denotes the set of leaves in  $\Phi$ . The (*leaf*) *size* of  $\Phi$  is defined as  $\text{size}(\Phi) := |\text{Leaves}(\Phi)| (= \frac{1}{2}(|\text{Sub}(\Phi)| + 1))$ . The *depth* of  $\Phi$  is its height as a tree (where a single leaf has depth 0).

## 3 Persistent Minterms

► **Notation 3.1**. For a partially ordered set  $L$  and  $m \in \mathbb{N}$ , we denote by  $\text{Seq}_{\leq}^m(L)$  the set of non-decreasing chains  $\vec{\lambda} = (\lambda_0, \lambda_1, \dots, \lambda_m)$  such that  $\lambda_0 \leq \lambda_1 \leq \dots \leq \lambda_m$ . (We will consistently index coordinates of  $\vec{\lambda}$  by  $\lambda_s, \lambda_t$  where  $0 \leq s \leq t \leq m$ .)

► **Notation 3.2**. For  $d, k \in \mathbb{N}$ , let  $\langle d \rangle_k := \binom{d+k-1}{k-1}$ . Note the identity  $\langle d \rangle_k = \langle d-1 \rangle_k + \langle d \rangle_{k-1}$ .



► **Lemma 3.1.** For all  $d, k \geq 1$  and  $\vec{a} \in \text{Seq}_{\leq}^k(\mathbb{R})$ , if  $a_k - a_0 > \langle \binom{d}{k} \rangle$ , then  $a_j - a_{j-1} > \langle \binom{d-1}{j} \rangle$  for some  $j \in \{1, \dots, k\}$ .

**Proof.** By induction on  $k$ : assuming  $a_k - a_0 > \langle \binom{d}{k} \rangle$ , either  $a_k - a_{k-1} > \langle \binom{d-1}{k} \rangle$ , in which case the lemma is satisfied with  $j = k$ , or else  $a_{k-1} - a_0 = (a_k - a_0) - (a_k - a_{k-1}) > \langle \binom{d}{k} \rangle - \langle \binom{d-1}{k} \rangle = \langle \binom{d}{k-1} \rangle$ , in which case we use the induction hypothesis for  $(a_0, \dots, a_{k-1}) \in \text{Seq}_{\leq}^{k-1}(\mathbb{R})$ . ◀

By the same basic induction, we have:

► **Lemma 3.2.** For all  $d, m \geq 1$  and  $\vec{x} \in \text{Seq}_{\leq}^m(\{0, 1\}^n)$ , if  $m \geq \langle \binom{d}{|x_m|} \rangle$ , then  $x_s = x_t$  for some  $0 \leq s \leq t \leq m$  with  $t - s \geq \langle \binom{d-1}{|x_s|} \rangle$ .

**Proof.** Suppose  $m \geq \langle \binom{d}{|x_m|} \rangle$  and let  $\ell := \min\{s \geq 0 : |x_s| = |x_m|\}$ . If  $m - \ell \geq \langle \binom{d-1}{|x_m|} \rangle$ , then we are done. Otherwise,  $\ell - 1 = (m - 1) - (m - \ell) \geq \langle \binom{d}{|x_m|} \rangle - 1 = \langle \binom{d-1}{|x_m|} \rangle - 1 \geq \langle \binom{d}{|x_{\ell-1}|} \rangle \geq \langle \binom{d}{|x_{\ell-1}|} \rangle$  and we use the induction hypothesis for the truncated sequence  $(x_0, \dots, x_{\ell-1}) \in \text{Seq}_{\leq}^{\ell-1}(\{0, 1\}^n)$ . ◀

► **Definition 3.3 (Persistent Minterms).** For  $\vec{f} \in \text{Seq}_{\leq}^m(\mathbb{B}_n^+)$  and  $x \in \{0, 1\}^n$ , we say that  $x$  is a  $d$ -persistent minterm of  $\vec{f}$  if it is a common minterm of  $f_s$  and  $f_t$  (i.e.  $x \in \mathcal{M}(f_s) \cap \mathcal{M}(f_t)$ ) for some  $0 \leq s \leq t \leq m$  such that  $t - s \geq \langle \binom{d}{|x|} \rangle$ . The set of  $d$ -persistent minterms of  $\vec{f}$  is denoted by  $\mathcal{M}_d(\vec{f})$ .

We have defined persistent minterms in a general way for sequences  $f_0 \leq f_1 \leq \dots \leq f_m$  of monotone functions. However, we will be interested in the persistent minterms of an *individual* monotone function  $f$  under a sequence  $\rho_0 \leq \rho_1 \leq \dots \leq \rho_d$  of monotone restrictions. (Eventually, we will utilize this notion by choosing *random* restrictions  $\vec{\rho}$ .)

► **Notation 3.3.** For  $f \in \mathbb{B}_n^+$  and  $\vec{\rho} \in \text{Seq}_{\leq}^m(\{0, 1\}^n)$ , let  $\mathcal{M}_d^{\vec{\rho}}(f) := \mathcal{M}_d(f^{\vee \rho_0} \leq f^{\vee \rho_1} \leq \dots \leq f^{\vee \rho_m})$ .

► **Lemma 3.4 (Persistent Minterm Lemma).** For all  $f, g \in \mathbb{B}_n^+$  and  $\vec{\rho} \in \text{Seq}_{\leq}^m(\{0, 1\}^n)$  and  $d, m \geq 1$ ,

$$\mathcal{M}_d^{\vec{\rho}}(f \vee g) \subseteq \mathcal{M}_{d-1}^{\vec{\rho}}(f) \cup \mathcal{M}_{d-1}^{\vec{\rho}}(g), \quad (3)$$

$$\mathcal{M}_d^{\vec{\rho}}(f \wedge g) \subseteq \{x \vee y : x \in \mathcal{M}_{d-1}^{\vec{\rho}}(f), y \in \mathcal{M}_{d-1}^{\vec{\rho}}(g)\}. \quad (4)$$

The proof, which we include in Appendix A, is straightforward (in particular, we show (4) using Lemma 3.2). We will return to persistent minterms in Section 5.2.

## 4 Average-Case $k$ -CYCLE

We depart from the setting of monotone functions  $\{0, 1\}^n \rightarrow \{0, 1\}$  (on  $n$  variables) and instead consider a domain  $\mathcal{G} \cong \{0, 1\}^{k^2 n}$  of graphs (with  $kn^2$  possible edges). Before defining  $\mathcal{G}$ , let us first clarify the role of  $k$ :

► **Definition 4.1.** Throughout the rest of this paper, let  $k = k(n) \in \mathbb{N}$  be an arbitrary parameter (i.e. function of  $n$ ) subject to  $k \leq \log \log n$ .

The constraint  $k \leq \log \log n$  is due to the factor of  $(1/2)^{O(2^k)}$  in Theorem 5.8. Outside of this theorem, all other lemmas in this paper hold for a larger range of  $k$ .



► **Definition 4.2** (*K-Partite Graphs*). All graphs in this paper are finite directed graphs without isolated vertices. Formally, a *graph* is a pair  $G = (V_G, E_G)$  where  $V_G$  is a finite set and  $E_G \subseteq V_G \times V_G$  and  $V_G = \bigcup_{vw \in E_G} \{v, w\}$ . As a special case,  $\emptyset$  denotes the *empty graph* with  $V_\emptyset = E_\emptyset = \emptyset$  (the empty set).  $K$  denotes the *k-cycle graph* with  $V_K = \{v_0, v_1, \dots, v_{k-1}\}$  and  $E_K = \{v_0v_1, v_1v_2, \dots, v_{k-1}v_0\}$ . (We never write these indices explicitly, instead always writing  $v \in V_K, vw \in E_K$  or  $e \in E_K$ .) We denote by  $\mathcal{G}$  ( $= \mathcal{G}(k, n)$ ) the set of *K-partite graphs*  $G$  satisfying

$$V_G \subseteq \{v^{(i)} : v \in V_K, i \in [n]\}, \quad E_G \subseteq \{v^{(i)}w^{(j)} : vw \in E_K, i, j \in [n]\}.$$

Here  $v^{(i)}$  and  $v^{(i)}w^{(j)}$  are just a friendly notation for ordered pairs  $(v, i)$  and  $((v, i), (w, j))$ . In the context of functions  $\mathcal{G} \rightarrow \{0, 1\}$ , we identify  $\mathcal{G}$  with the hypercube  $\{0, 1\}^{kn^2}$ .

► **Definition 4.3** (*k-CYCLE*). For  $G \in \mathcal{G}$ , we say that  $G$  is a *k-cycle* if  $G$  is isomorphic to  $K$ . Note that  $G$  is a *k-cycle* if and only if there exists a function  $\iota : V_K \rightarrow [n]$  such that  $E_G = \{v^{\iota(v)}w^{\iota(w)} : vw \in E_K\}$ . We say that  $G$  has a *k-cycle* if it contains a *k-cycle* as a subgraph. *k-CYCLE* denotes the monotone function  $\mathcal{G} \rightarrow \{0, 1\}$  which takes value 1 on  $G$  if, and only if,  $G$  has a *k-cycle*.

We are interested in the average-case analysis of *k-CYCLE*. For this purpose, we define three random graphs needed to state our main lemma (on the noise-invariance of minterms of *k-CYCLE*).

► **Definition 4.4** (*Random Graphs  $\Gamma, \circlearrowleft$  and  $\Xi_\ell$* ). Let  $\Gamma, \circlearrowleft$  and  $\Xi_\ell$  be the following (independent) random graphs in  $\mathcal{G}$ :

- Let  $\Gamma$  be the (*K-partite, Erdős-Rényi*) random graph in  $\mathcal{G}$  which includes each potential edge independently with probability  $p$  (i.e.  $\mathbb{P}[\Gamma = G] = p^{|E_G|}(1-p)^{kn^2-|E_G|}$  where  $p = p(k, n) (\sim (\ln 2)^{1/k}/n)$  is the unique “ $\frac{1}{2}$ -threshold” such that  $\mathbb{P}[k\text{-CYCLE}(\Gamma) = 1] = \frac{1}{2}$ ).
- Let  $\circlearrowleft$  be a uniform random *k-cycle* in  $\mathcal{G}$ . For  $e \in E_K$ , we write  $\circlearrowleft^{-e}$  for the graph obtained from  $\circlearrowleft$  by deleting the edge in  $\circlearrowleft$  corresponding to  $e$ . Note that  $\circlearrowleft^{-e}$  is a path of length  $k - 1$ .
- For  $\ell \in \mathbb{N}$ , let  $\Xi_\ell$  be the random graph  $\circlearrowleft_1^{-e_1} \cup \dots \cup \circlearrowleft_\ell^{-e_\ell}$  where  $\circlearrowleft_1, \dots, \circlearrowleft_\ell$  are uniform random *k-cycles* and  $e_1, \dots, e_\ell$  are uniform random edges in  $E_K$ . Equivalently,  $\Xi_\ell$  is the union of  $\ell$  uniform random paths of length  $k - 1$ .

We will only consider values of  $\ell$  much less than  $\sqrt{n}$ , where random paths  $\circlearrowleft_1^{-e_1} \cup \dots \cup \circlearrowleft_\ell^{-e_\ell}$  are likely to be vertex-disjoint. The letter  $\Xi$  is mnemonic for this situation.

We state the key lemma of this section, whose proof is included in the full paper.

► **Lemma 4.5.** *For every monotone function  $f : \mathcal{G} \rightarrow \{0, 1\}$  and  $\ell \in \mathbb{N}$ , if*

$$\mathbb{P}_\Gamma [f(\Gamma) = k\text{-CYCLE}(\Gamma)] \geq \frac{1}{2} + \frac{C(\ell + 1)k^2}{\sqrt{n}} \tag{5}$$

where  $C > 0$  is a universal constant, then there exists  $G \in \mathcal{G}$  such that

$$\mathbb{P}_{\Xi_\ell} \left[ \mathbb{P}_{\circlearrowleft} [\circlearrowleft \in \mathcal{M}(f^{\cup G}) \cap \mathcal{M}(f^{\cup G \cup \Xi_\ell})] \geq n^{-1/2} \right] \geq n^{-1/2}. \tag{6}$$

Lemma 4.5 says the following: (in the case  $\ell = 0$ ) if a monotone function  $f$  has correlation  $\gg k^2/\sqrt{n}$  with *k-CYCLE* on  $\Gamma$ , then there exists a graph  $G$  such that a non-negligible fraction of *k-cycles* are minterms of  $f^{\cup G}$ . Moreover, (for  $\ell \geq 1$ ) if this correlation is  $\gg \ell k^2/\sqrt{n}$ , then these minterms are “ $\Xi_\ell$ -noise-invariant” in the following sense: with probability  $\geq n^{-1/2}$  over  $\Xi_\ell$ , at least  $1/\sqrt{n}$  fraction of *k-cycles* are common minterms of  $f^{\cup G}$  and  $f^{\cup G \cup \Xi_\ell}$ .

The tie-in to persistent minterms is clear. Let  $d \in \mathbb{N}$  and suppose  $\ell$  is a multiple of  $m := \binom{d}{k}$ . We may generate  $\Xi_\ell$  as a union of independent  $\Xi_{\ell/m}^{(1)}, \dots, \Xi_{\ell/m}^{(m)}$ . Writing  $\rho_s$  for the partial union  $\Xi_{\ell/m}^{(1)} \cup \dots \cup \Xi_{\ell/m}^{(s)}$ , we have a non-decreasing sequence  $\vec{\rho} \in \text{Seq}_{\leq}^m(\mathcal{G})$ . Notice that every  $k$ -cycle which is a common minterm in  $\mathcal{M}(f^{\cup G}) \cap \mathcal{M}(f^{\cup G \cup \Xi_\ell})$  is a  $d$ -persistent minterm in  $\mathcal{M}_d^{\vec{\rho}}(f^{\cup G})$ . (This observation shows up in the proof of Theorem 1.1 in Section 6.)

## 5 Pathset Complexity

### 5.1 The Basic Framework

We present the definitions required to state the pathset complexity lower bound (Theorem 5.8), which we use in our main theorem (Theorem 1.1). For background on these definitions (key examples, upper bounds, etc.), the reader is referred to the paper [56].

► **Definition 5.1** (Pattern Graphs). Subgraphs of  $K$  are called *pattern graphs* and designated by letters  $A, B, C$ . Recall that graphs (by definition in this paper) have no isolated vertices. Therefore, pattern graphs  $A \subseteq K$  are in one-to-one correspondence with subsets  $E_A \subseteq E_K$ .

An important parameter of pattern graphs  $A \subseteq K$  is the number  $|V_A| - |E_A|$ . Note that every pattern graph, other than  $K$  itself, is a disjoint union of paths. Therefore,

$$A \neq K \Rightarrow |V_A| - |E_A| = |\{\text{connected components of } A\}|. \quad (7)$$

Also note that  $0 \leq |V_A| - |E_A| \leq k/2$  and  $|V_A| - |E_A| = 0 \Leftrightarrow A \in \{\emptyset, K\}$ .

► **Definition 5.2** (Sections). For  $A \subseteq K$ , an  $A$ -*section* is a graph  $A' \in \mathcal{G}$  such that  $E_{A'} = \{v^{\iota(v)} w^{\iota(w)} : vw \in E_A\}$  for some function  $\iota : V_A \rightarrow [n]$ . (As a special case, the empty graph  $\emptyset$  is the unique  $\emptyset$ -section.) The set of all  $A$ -sections is denoted by  $\mathcal{G}_A$ . As a matter of notation, we consistently write  $A$ -sections using primes ( $A', A''$ , etc.) Every  $A' \in \mathcal{G}_A$  is isomorphic to  $A$  via the projection  $v^{(i)} \mapsto v$ .

We have already encountered  $K$ -sections and  $K \setminus \{e\}$ -sections in the guise of random graphs  $\circ$  and  $\circ^{-e}$ . (Note that  $K$ -sections are the same as  $k$ -cycles in  $\mathcal{G}$  (Def. 4.2).)

► **Definition 5.3** (Pathsets). For  $A \subseteq K$ , subsets of  $\mathcal{G}_A$  (i.e. sets of  $A$ -sections) are called *pathsets over  $A$* . As a special case, note that there are two distinct pathsets over  $\emptyset$ : the empty set  $\emptyset$  and the “identity” pathset  $\{\emptyset\}$ . Every non-empty pathset  $\mathcal{A}$  is a pathset over a unique  $A \subseteq K$ , which we call its *underlying pattern graph*. Pathsets over  $A, B, C, K$  are consistently designated by the respective calligraphic letters  $\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{K}$ . The *density* of a pathset  $\mathcal{A}$  is defined by

$$\text{density}(\mathcal{A}) := |\mathcal{A}| / n^{|V_A|} = \mathbb{P}_{A' \in \mathcal{G}_A} [A' \in \mathcal{A}]. \quad (8)$$

► **Definition 5.4** (Joins). For any two pathsets  $\mathcal{A}$  and  $\mathcal{B}$ , the *join*  $\mathcal{A} \bowtie \mathcal{B}$  is the pathset (over  $A \cup B$ ) defined by

$$\mathcal{A} \bowtie \mathcal{B} := \{C' \in \mathcal{G}_{A \cup B} : C' = A' \cup B' \text{ for some } A' \in \mathcal{A} \text{ and } B' \in \mathcal{B}\}. \quad (9)$$

Note that  $\bowtie$  is an associative, commutative and idempotent operation on pathsets. Moreover,  $\emptyset$  and  $\{\emptyset\}$  act as the zero and identity:  $\mathcal{A} \bowtie \emptyset = \emptyset$  and  $\mathcal{A} \bowtie \{\emptyset\} = \mathcal{A}$ . (Taking the view of a pathset  $\mathcal{A}$  as a “ $V_A$ -ary relation” (i.e. a subset of  $[n]^{V_A}$ ),  $\bowtie$  is the standard relational join operation.)

► **Definition 5.5** (Restrictions). For pathsets  $\mathcal{A}$  and  $\mathcal{B}$ , we say that  $\mathcal{B}$  is a *restriction* of  $\mathcal{A}$ , denoted  $\mathcal{B} \preceq \mathcal{A}$ , if  $B \subseteq A$  and there exists  $\overline{B'} \in \mathcal{G}_{A \setminus B}$  such that  $\mathcal{B} = \{B' \in \mathcal{G}_B : B' \cup \overline{B'} \in \mathcal{A}\}$ .  $\mathcal{B}$  is a *proper restriction* of  $\mathcal{A}$ , denoted  $\mathcal{B} \prec \mathcal{A}$ , if  $\mathcal{B} \preceq \mathcal{A}$  and  $\mathcal{B} \neq \mathcal{A}$ .

► **Definition 5.6** (Smallness). For  $\varepsilon > 0$ , a pathset  $\mathcal{A}$  is  $\varepsilon$ -*small* if it satisfies

$$\text{density}(\mathcal{B}) \leq \varepsilon^{|V_B| - |E_B|} \text{ for all } \mathcal{B} \preceq \mathcal{A}. \quad (10)$$

The set of  $\varepsilon$ -small pathsets (over all pattern graphs) is denoted by  $\mathcal{P}_\varepsilon$ .

Note that every pathset over  $\emptyset$  or  $K$  is  $\varepsilon$ -small, since  $|V_\emptyset| - |E_\emptyset| = |V_K| - |E_K| = 0$ .  $\varepsilon$ -smallness is obviously preserved under subsets, as well as under restrictions: if  $\mathcal{A} \in \mathcal{P}_\varepsilon$ , then  $\mathcal{A}_0 \in \mathcal{P}_\varepsilon$  and  $\mathcal{B} \in \mathcal{P}_\varepsilon$  for every  $\mathcal{A}_0 \subseteq \mathcal{A}$  and  $\mathcal{B} \preceq \mathcal{A}$ . Somewhat less obvious is the fact that  $\varepsilon$ -smallness is also preserved under joins (Lemma 5.5 of [56]): if  $\mathcal{A}, \mathcal{B} \in \mathcal{P}_\varepsilon$ , then  $\mathcal{A} \bowtie \mathcal{B} \in \mathcal{P}_\varepsilon$ .

► **Definition 5.7** (Pathset Complexity). For any  $\varepsilon > 0$  (“smallness parameter”), *pathset complexity* is the function  $\chi_\varepsilon : \mathcal{P}_\varepsilon \rightarrow \mathbb{N}$  defined inductively as follows:

- (base case)  $\chi_\varepsilon(\emptyset) = \chi_\varepsilon(\{\emptyset\}) = 0$  and  $\chi_\varepsilon(\mathcal{A}) = |\mathcal{A}|$  if  $|E_A| = 1$ ,
- (induction case) if  $|E_A| \geq 2$ , then

$$\chi_\varepsilon(\mathcal{A}) := \min_{(\mathcal{B}_i, \mathcal{C}_i)_i} \sum_i \chi_\varepsilon(\mathcal{B}_i) + \chi_\varepsilon(\mathcal{C}_i)$$

where  $(\mathcal{B}_i, \mathcal{C}_i)_i$  ranges over all sequences of  $\varepsilon$ -small pathsets  $\mathcal{B}_i, \mathcal{C}_i \in \mathcal{P}_\varepsilon$  such that  $\mathcal{B}_i, \mathcal{C}_i \not\subseteq \mathcal{A}$  and  $\mathcal{B}_i \cup \mathcal{C}_i = \mathcal{A}$  and  $\mathcal{A} \subseteq \bigcup_i \mathcal{B}_i \bowtie \mathcal{C}_i$ .

In other words, for the (induction case) we consider all possible *coverings* of  $\mathcal{A}$  by joins of  $\varepsilon$ -small pathsets over *proper* subgraphs of  $A$ .

It is clear from this definition that pathset complexity satisfies the following inequalities:

- (monotonicity)  $\chi_\varepsilon(\mathcal{A}_1) \leq \chi_\varepsilon(\mathcal{A}_2)$  for all  $\mathcal{A}_1 \subseteq \mathcal{A}_2 \in \mathcal{P}_\varepsilon$ ,
- (sub-additivity)  $\chi_\varepsilon(\mathcal{A}_1 \cup \mathcal{A}_2) \leq \chi_\varepsilon(\mathcal{A}_1) + \chi_\varepsilon(\mathcal{A}_2)$  for all  $\mathcal{A}_1, \mathcal{A}_2$  such that  $\mathcal{A}_1 \cup \mathcal{A}_2 \in \mathcal{P}_\varepsilon$ ,
- (join inequality)  $\chi_\varepsilon(\mathcal{A} \bowtie \mathcal{B}) \leq \chi_\varepsilon(\mathcal{A}) + \chi_\varepsilon(\mathcal{B})$  for all  $\mathcal{A}, \mathcal{B} \in \mathcal{P}_\varepsilon$ .

In fact, these three inequalities provide a *dual characterization* of pathset complexity:  $\chi_\varepsilon$  is the unique pointwise maximal function  $\mathcal{P}_\varepsilon \rightarrow \mathbb{N}$  which satisfies (base case), (monotonicity), (sub-additivity) and (join inequality).

The following lower bound on pathset complexity was shown in [56]:

► **Theorem 5.8** (Pathset Complexity Lower Bound). *For every pathset  $\mathcal{K}$  over  $K$ ,*

$$\chi_\varepsilon(\mathcal{K}) \geq (1/2)^{O(2^k)} \cdot (1/\varepsilon)^{\frac{1}{8} \log k} \cdot \text{density}(\mathcal{K}). \quad (11)$$

Theorem 5.8 corresponds to Theorem 5.8 of [56]. We remark that the lower bound proved in [56] applies more broadly to pathsets  $\mathcal{A} \in \mathcal{P}_\varepsilon$  over any pattern graph  $A \subseteq K$ :

$$\chi_\varepsilon(\mathcal{A}) \geq (1/2)^{O(2^{|E_A|})} \cdot (1/\varepsilon)^{\frac{1}{8} \log(\text{length}(A)) + |V_A| - |E_A|} \cdot \text{density}(\mathcal{A}) \quad (12)$$

where  $\text{length}(A)$  equals the number of edges in the largest connected component of  $A$ . In fact, (12) follows from an even more general lower bound for *pathset complexity with respect to patterns* (Theorem 8.3 of [56]). However, for the application in this paper, we only require the bound (11) for pathsets over  $K$ .

## 5.2 Pathsets of Persistent Minterms

In order to prove formula-size lower bounds using pathset complexity, we associate pathsets with all monotone formulas on  $kn^2$  variables. The pathsets need to satisfy certain consistency conditions; moreover, these (random) pathsets must be  $\varepsilon$ -small (with high probability). Persistent minterms and random restrictions  $\Xi_\ell$  accomplish both of these goals. In this subsection, we show how to define appropriate pathsets using persistent minterms; we deal with  $\varepsilon$ -smallness in the next subsection.

► **Definition 5.9** (Pathsets  $\mathcal{M}_A(f)$  and  $\mathcal{P}_A^{\vec{\rho}}(\Phi)$ ). For a monotone function  $f : \mathcal{G} \rightarrow \{0, 1\}$  and  $A \subseteq K$ , let  $\mathcal{M}_A(f) := \mathcal{G}_A \cap \mathcal{M}(f)$  be the pathset of  $A$ -sections which are minterms of  $f$ . For a monotone formula  $\Phi$  and  $\vec{\rho} \in \text{Seq}_{\leq}^m(\mathcal{G})$  and  $A \subseteq K$ , the pathset  $\mathcal{P}_A^{\vec{\rho}}(\Phi)$  (over  $A$ ) is defined by

$$\mathcal{P}_A^{\vec{\rho}}(\Phi) := \mathcal{G}_A \cap \mathcal{M}_{\text{depth}(\Phi)}^{\vec{\rho}}(\Phi). \quad (13)$$

That is, the  $\mathcal{P}_A^{\vec{\rho}}(\Phi)$  is the set of  $A$ -sections which are  $\text{depth}(\Phi)$ -persistent minterms of  $\Phi$  under the sequence  $\vec{\rho}$ .

Unpacking definitions, for all  $A \neq \emptyset$ , we have the expression

$$\mathcal{P}_A^{\vec{\rho}}(\Phi) = \bigcup_{\substack{0 \leq s \leq t \leq m: \\ t-s \geq \binom{\text{depth}(\Phi)}{|E_A|}}} (\mathcal{M}_A(\Phi^{\cup \rho_s}) \cap \mathcal{M}_A(\Phi^{\cup \rho_t})) \subseteq \bigcup_{0 \leq s \leq m-1} (\mathcal{M}_A(\Phi^{\cup \rho_s}) \cap \mathcal{M}_A(\Phi^{\cup \rho_{s+1}})). \quad (14)$$

The following lemma is a straightforward consequence of (14).

► **Lemma 5.10.** *If  $\mathcal{P}_A^{\vec{\rho}}(\Phi)$  is not  $\varepsilon$ -small, then  $\mathcal{M}_A(\Phi^{\cup \rho_s}) \cap \mathcal{M}_A(\Phi^{\cup \rho_{s+1}})$  is not  $(\varepsilon/m)$ -small for some  $s \in \{0, \dots, m-1\}$ .*

**Proof.** Assume  $\mathcal{P}_A^{\vec{\rho}}(\Phi)$  is not  $\varepsilon$ -small. By Def. 5.6, there exists a restriction  $\mathcal{B} \preceq \mathcal{P}_A^{\vec{\rho}}(\Phi)$  such that  $\text{density}(\mathcal{B}) > \varepsilon^{|V_B| - |E_B|}$ . (Note that  $A, B \notin \{\emptyset, K\}$ .) By Def. 5.5, there exists an  $(A \setminus B)$ -section  $\overline{B'} \in \mathcal{G}_{A \setminus B}$  such that  $\mathcal{B} = \{B' \in \mathcal{G}_B : B' \cup \overline{B'} \in \mathcal{P}_A^{\vec{\rho}}(\Phi)\}$ . Writing  $\mathcal{A}_s$  for  $\mathcal{M}_A(\Phi^{\cup \rho_s}) \cap \mathcal{M}_A(\Phi^{\cup \rho_{s+1}})$ , we have  $\mathcal{P}_A^{\vec{\rho}}(\Phi) \subseteq \bigcup_{s=0}^{m-1} \mathcal{A}_s$  by (14), hence  $\mathcal{B} \subseteq \bigcup_{s=0}^{m-1} \{B' \in \mathcal{G}_B : B' \cup \overline{B'} \in \mathcal{A}_s\}$ . It follows that there exists  $s \in \{0, \dots, m-1\}$  such that

$$\text{density}(\{B' \in \mathcal{G}_B : B' \cup \overline{B'} \in \mathcal{A}_s\}) \geq \text{density}(\mathcal{B})/m > \varepsilon^{|V_B| - |E_B|}/m \geq (\varepsilon/m)^{|V_B| - |E_B|}.$$

Since  $\{B' \in \mathcal{G}_B : B' \cup \overline{B'} \in \mathcal{A}_s\} \preceq \mathcal{A}_s$ , we conclude that  $\mathcal{A}_s$  is not  $(\varepsilon/m)$ -small. ◀

We next restate the Persistent Minterm Lemma 3.4 in terms of pathsets  $\mathcal{P}_A^{\vec{\rho}}(\Phi)$ .

► **Lemma 5.11.** *For all monotone functions  $f, g$  and monotone formulas  $\Phi, \Psi$  and  $\vec{\rho} \in \text{Seq}_{\leq}^m(\mathcal{G})$  and  $A \subseteq K$ ,*

$$\mathcal{P}_A^{\vec{\rho}}(\Phi \vee \Psi) \subseteq \mathcal{P}_A^{\vec{\rho}}(\Phi) \cup \mathcal{P}_A^{\vec{\rho}}(\Psi), \quad (15)$$

$$\mathcal{P}_A^{\vec{\rho}}(\Phi \wedge \Psi) \subseteq \bigcup_{B, C \subseteq A : B \cup C = A} \mathcal{P}_B^{\vec{\rho}}(\Phi) \bowtie \mathcal{P}_C^{\vec{\rho}}(\Psi). \quad (16)$$

The main lemma of this subsection gives the key relationship between pathset complexity and formula size and depth.

► **Lemma 5.12.** *Suppose  $\Phi$  is a monotone formula and  $\vec{\rho} \in \text{Seq}_{\leq}^m(\mathcal{G})$  such that pathsets  $\mathcal{P}_A^{\vec{\rho}}(\Psi)$  are  $\varepsilon$ -small for all  $\Psi \in \text{Sub}(\Phi)$  and  $A \subseteq K$ . Then*

$$\chi_\varepsilon(\mathcal{P}_K^{\vec{\rho}}(\Phi)) \leq 2^{O(k^2)} \cdot \text{depth}(\Phi)^k \cdot \text{size}(\Phi). \quad (17)$$

Although the statement of Lemma 5.12 might appear complicated, the proof is actually quite simple. The derivation of (17) uses only Lemma 5.11 and the key properties (monotonicity), (sub-additivity) and (join inequality) of pathset complexity. The proof of Lemma 5.12, which is essentially the same as Lemma 6.7 in [56], is included in Appendix B.

### 5.3 Smallness Lemma

In the last subsection, we defined pathsets  $\mathcal{P}_A^{\vec{\rho}}(\Phi)$  (for an arbitrary sequence  $\vec{\rho} \in \text{Seq}_{\leq}^m(\mathcal{G})$ ) and showed a relationship between pathset complexity and formula size under the condition that all of the relevant pathsets are  $\varepsilon$ -small. The next lemma give a means of establishing  $\varepsilon$ -smallness.

► **Lemma 5.13.** *For every monotone function  $f : \mathcal{G} \rightarrow \{0, 1\}$  and  $A \subseteq K$  and  $\ell \in \mathbb{N}$  and  $\varepsilon > 0$ ,*

$$\mathbb{P}_{\Xi_\ell} [\mathcal{M}_A(f) \cap \mathcal{M}_A(f^{\cup \Xi_\ell}) \text{ is \underline{not} } \varepsilon\text{-small}] \leq (2n)^k \cdot \exp(-\Omega(\varepsilon \ell / k^2)). \quad (18)$$

The main tools in the proof of Lemma 5.13 (included in the full version of this paper) are Janson's Inequality [33] and the sunflower-plucking technique of Razborov [51].

## 6 Proof of Theorem 1.1 (Correlation Bound for $k$ -CYCLE)

**Proof of Theorem 1.1.** Let  $k \leq \log \log n$  and suppose  $\Phi$  is a monotone formula such that

$$\mathbb{P}_\Gamma [f(\Gamma) = k\text{-CYCLE}(\Gamma)] = \frac{1}{2} + n^{-1/2+c}.$$

Our goal is to show the lower bound  $\text{size}(\Phi) = n^{\Omega(c \log k)}$ .

Using the fact that  $n^{O(\log k)}$  is an upper bound on the size of monotone formulas for  $k$ -CYCLE (together with the “formula balancing lemma” [59, 66]: every monotone formula of size  $S$  is equivalent to a monotone formula of depth  $O(\log S)$ ) we may assume that  $\text{size}(\Phi) = n^{O(\log k)}$  and  $\text{depth}(\Phi) = O(\log k \cdot \log n)$ . However, for purposes of this proof, it is enough for us to assume much weaker upper bounds  $\text{size}(\Phi) \leq \exp(n^{1/k})$  and  $\text{depth}(\Phi) \leq n^{1/k}$ . We also assume  $c = \Omega(1/\log k)$ , since otherwise there is nothing to prove.

We set parameters  $m, \ell, \varepsilon$  as follows:

$$m := \langle \text{depth}(\Phi) \rangle_k (= \langle \text{depth}(\Phi) + k - 1 \rangle_{k-1}), \quad \ell := n^{c/2}, \quad \varepsilon := n^{-c/4}. \quad (19)$$

Note that  $m = O(\text{depth}(\Phi))^k = n^{o(c)}$ . We have  $n^{-1/2+c} = \omega((m\ell + 1)k^2/\sqrt{n})$ , that is,  $\Phi$  satisfies the hypothesis (5) of Lemma 4.5 (for all sufficiently large  $n$ ). Therefore, by Lemma 4.5, there exists  $G \in \mathcal{G}$  such that

$$\mathbb{P}_{\Xi_{\ell m}} \left[ \mathbb{P}_{\odot} [\odot \in \mathcal{M}(\Phi^{\cup G}) \cap \mathcal{M}(\Phi^{\cup G \cup \Xi_{m\ell}})] \geq n^{-1/2} \right] = \Omega(n^{-1/2}). \quad (20)$$

Fixing any such  $G$ , we now generate *random*  $\vec{\rho} \in \text{Seq}_{\leq}^m(\mathcal{G})$  as follows:

- Let  $\Xi_\ell^{(1)}, \dots, \Xi_\ell^{(m)}$  be independent random copies of  $\Xi_\ell$ .
- For  $s \in \{0, \dots, m\}$ , let  $\rho_s := G \cup (\Xi_\ell^{(1)} \cup \dots \cup \Xi_\ell^{(s)})$ .

By our choice of  $m = \langle \text{depth}(\Phi) \rangle_k$  and Def. 5.9 of  $\mathcal{P}_K^{\vec{\rho}}(\Phi)$  (see (14)), we have

$$\mathcal{P}_K^{\vec{\rho}}(\Phi) = \mathcal{M}_K(\Phi^{\cup \rho_0}) \cap \mathcal{M}_K(\Phi^{\cup \rho_m}).$$

Since  $\circlearrowleft$  is uniform in  $\mathcal{G}_K$ , it follows (by definition (8) of  $\text{density}(\cdot)$ ) that

$$\text{density}(\mathcal{P}_K^{\vec{\rho}}(\Phi)) = \mathbb{P}_{\circlearrowleft} [\circlearrowleft \in \mathcal{M}(\Phi^{\cup \rho_0}) \cap \mathcal{M}(\Phi^{\cup \rho_m})].$$

Since  $\rho_0 = G$  and  $\rho_m \stackrel{d}{=} G \cup \Xi_{m\ell}$ , we see that (20) is equivalent to

$$\mathbb{P}_{\vec{\rho}} [\text{density}(\mathcal{P}_K^{\vec{\rho}}(\Phi)) \geq n^{-1/2}] = \Omega(n^{-1/2}). \tag{21}$$

We next observe that, with all-but-negligible probability  $1 - n^{-\omega(1)}$ , pathsets  $\mathcal{P}_A^{\vec{\rho}}(\Psi)$  are all  $\varepsilon$ -small:

$$\begin{aligned} \mathbb{P}_{\vec{\rho}} \left[ \bigvee_{\Psi \in \text{Sub}(\Phi)} \bigvee_{\emptyset \subset A \subset K} \mathcal{P}_A^{\vec{\rho}}(\Psi) \text{ is \underline{not} } \varepsilon\text{-small} \right] & \quad (\text{Lemma 5.10}) \tag{22} \\ & \leq \sum_{\Psi \in \text{Sub}(\Phi)} \sum_{\emptyset \subset A \subset K} \sum_{0 \leq s \leq m-1} \mathbb{P}_{\vec{\rho}} [\mathcal{M}_A(\Psi^{\cup \rho_s}) \cap \mathcal{M}_A(\Psi^{\cup \rho_{s+1}}) \text{ is \underline{not} } (\varepsilon/m)\text{-small}] \\ & \leq \text{size}(\Phi) \cdot 2^k \cdot m \cdot \exp(-\Omega(\varepsilon \ell / k^2 m)) \quad (\text{Lemma 5.13}) \\ & = \exp(O(n^{1/k})) \cdot \exp(-n^{c/4 - o(c)}) \quad (\text{size}(\Phi) \leq \exp(n^{1/k})) \\ & = n^{-\omega(1)} \quad (c = \Omega(1/\log k)). \end{aligned}$$

As the upshot of (21) and (22), (for all sufficiently large  $n$ ) there exists  $\vec{\rho} \in \text{Seq}_{\leq}^m(\mathcal{G})$  satisfying both

- **Dense**( $\vec{\rho}$ ), the event that  $\text{density}(\mathcal{P}_K^{\vec{\rho}}(\Phi)) \geq n^{-1/2}$ , and
- **Small**( $\vec{\rho}$ ), the event that pathsets  $\mathcal{P}_A^{\vec{\rho}}(\Psi)$  are  $\varepsilon$ -small for all  $\Psi \in \text{Sub}(\Phi)$  and  $A \subseteq K$ .

Fixing any such  $\vec{\rho}$ , we complete the reduction to our pathset complexity lower bound (using  $k \leq \log \log n$ ):

$$\begin{aligned} \text{size}(\Phi) & \geq \text{depth}(\Phi)^{-k} \cdot 2^{-O(k^2)} \cdot \chi_{\varepsilon}(\mathcal{P}_K^{\vec{\rho}}(\Phi)) \quad (\text{Small}(\vec{\rho}) \text{ and Lemma 5.12}) \\ & \geq n^{-O(1)} \cdot \chi_{\varepsilon}(\mathcal{P}_K^{\vec{\rho}}(\Phi)) \quad (\text{depth}(\Phi) \leq n^{1/k}) \\ & \geq n^{-O(1)} \cdot 2^{-O(2^k)} \cdot (1/\varepsilon)^{\frac{1}{6} \log k} \cdot \text{density}(\mathcal{P}_K^{\vec{\rho}}(\Phi)) \quad (\text{Theorem 5.8}) \\ & = n^{(c/24) \log k - O(1)} \quad (\text{Dense}(\vec{\rho})). \end{aligned}$$

Therefore,  $\text{size}(\Phi) = n^{\Omega(c \log k)}$  as required. ◀

**Acknowledgements.** I would like to thank Li-Yang Tan for valuable feedback on various technical aspects of this paper, Rahul Santhanam and Osamu Watanabe for helpful discussions, Siyao Guo for debugging a confusing typo, and the anonymous referees for many useful suggestions.

---

**References**

- 1 Miklós Ajtai.  $\Sigma_1^1$  formulae on finite structures. *Annals of Pure and Applied Logic*, 24:1–48, 1983.
- 2 Miklós Ajtai and Yuri Gurevich. Monotone versus positive. *J. ACM*, 34:1004–1015, 1987.
- 3 Miklós Ajtai, János Komlós, and Endre Szemerédi. An  $O(n \log n)$  sorting network. In *Proceedings of the 15th Annual ACM Symposium on Theory of Computing*, pages 1–9. ACM, 1983.

- 4 Noga Alon and Ravi B. Boppana. The monotone circuit complexity of boolean functions. *Combinatorica*, 7(1):1–22, 1987.
- 5 Noga Alon and Joel Spencer. *The Probabilistic Method, 3rd Edition*. John Wiley, 2008.
- 6 Kazuyuki Amano and Akira Maruoka. Potential of the approximation method. In *Foundations of Computer Science, 1996. Proceedings., 37th Annual Symposium on*, pages 431–440. IEEE, 1996.
- 7 Kazuyuki Amano and Akira Maruoka. A superpolynomial lower bound for a circuit computing the clique function with at most  $(1/6) \log \log n$  negation gates. *SIAM Journal on Computing*, 35(1):201–216, 2005.
- 8 Alexander E. Andreev. On a method for obtaining lower bounds for the complexity of individual monotone functions. 31(3):530–534, 1985.
- 9 Alexander E. Andreev, Andrea E.F. Clementi, and José D.P. Rolim. Optimal bounds for the approximation of boolean functions and some applications. *Theoretical Computer Science*, 180(1):243–268, 1997.
- 10 Richard Arratia, Larry Goldstein, and Louis Gordon. Poisson approximation and the chen-stein method. *Statistical Science*, pages 403–424, 1990.
- 11 Robert Beals, Tetsuro Nishino, and Keisuke Tanaka. On the complexity of negation-limited boolean networks. *SIAM Journal on Computing*, 27(5):1334–1347, 1998.
- 12 Shai Ben-David, Benny Chor, Oded Goldreich, and Michel Luby. On the theory of average case complexity. *Journal of Computer and System Sciences*, 44(2):193–219, 1992.
- 13 Michael Ben-Or and Nathan Linial. Collective coin flipping. *Randomness and Computation*, 5:91–115, 1990.
- 14 Stuart J. Berkowitz. On some relationships between monotone and nonmonotone circuit complexity. Technical report, Department of Computer Science, University of Toronto, Canada, Toronto, Canada, 1982.
- 15 Avrim Blum, Carl Burch, and John Langford. On learning monotone boolean functions. In *Foundations of Computer Science, 1998. Proceedings. 39th Annual Symposium on*, pages 408–415. IEEE, 1998.
- 16 Norbert Blum. On negations in boolean networks. In *Efficient Algorithms*, pages 18–29. Springer, 2009.
- 17 Andrej Bogdanov and Luca Trevisan. Average-case complexity. *Theoretical Computer Science*, 2(1):1–106, 2006.
- 18 Béla Bollobás and A.G. Thomason. Threshold functions. *Combinatorica*, 7(1):35–38, 1987.
- 19 Nader H. Bshouty and Christino Tamon. On the Fourier spectrum of monotone functions. *Journal of the ACM (JACM)*, 43(4):747–770, 1996.
- 20 Yuval Filmus, Toniann Pitassi, Robert Robere, and Stephen A Cook. Average case lower bounds for monotone switching networks. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 20, page 54, 2013.
- 21 Michael J. Fischer. The complexity of negation-limited networks – a brief survey. In *Automata Theory and Formal Languages 2nd GI Conference Kaiserslautern, May 20–23, 1975*, pages 71–82. Springer, 1975.
- 22 Cees M. Fortuin, Pieter W. Kasteleyn, and Jean Ginibre. Correlation inequalities on some partially ordered sets. *Communications in Mathematical Physics*, 22(2):89–103, 1971.
- 23 Ehud Friedgut. Sharp thresholds of graph properties, and the  $k$ -SAT problem. *J. Amer. Math. Soc.*, 12:1017–1054, 1998.
- 24 Merrick L. Furst, James B. Saxe, and Michael Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical Systems Theory*, 17:13–27, 1984.
- 25 Mika Göös and Toniann Pitassi. Communication lower bounds via critical block sensitivity. *arXiv preprint arXiv:1311.2355*, 2013.



- 26 Michelangelo Grigni and Michael Sipser. Monotone complexity. *Boolean function complexity*, 169:57–75, 1992.
- 27 Michelangelo Grigni and Michael Sipser. Monotone separation of logarithmic space from logarithmic depth. *Journal of Computer and System Sciences*, 50(3):433–437, 1995.
- 28 Johan Håstad. The shrinkage exponent of de Morgan formulas is 2. *SIAM Journal on Computing*, 27(1):48–64, 1998.
- 29 Johan Håstad. On the correlation of parity and small-depth circuits. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 19, page 137, 2012.
- 30 Richard Holley. Remarks on the FKG inequalities. *Communications in Mathematical Physics*, 36(3):227–231, 1974.
- 31 Russell Impagliazzo. Hard-core distributions for somewhat hard problems. In *Foundations of Computer Science, 1995. Proceedings., 36th Annual Symposium on*, pages 538–545. IEEE, 1995.
- 32 Kazuo Iwama and Hiroki Morizumi. An explicit lower bound of  $5n - o(n)$  for boolean circuits. In *Mathematical foundations of computer science 2002*, pages 353–364. Springer, 2002.
- 33 Svante Janson. Poisson approximation for large deviations. *Random Structures & Algorithms*, 1(2):221–229, 1990.
- 34 Stasys Jukna. On the minimum number of negations leading to super-polynomial savings. *Information processing letters*, 89(2):71–74, 2004.
- 35 Stasys Jukna. *Boolean Function Complexity: Advances and Frontiers*, volume 27. Springer-Verlag Berlin Heidelberg, 2012.
- 36 George Karakostas, Jeff Kinne, and Dieter van Melkebeek. On derandomization and average-case complexity of monotone functions. *Theoretical Computer Science*, 434:35–44, 2012.
- 37 Mauricio Karchmer and Avi Wigderson. Monotone circuits for connectivity require super-logarithmic depth. *SIAM Journal on Discrete Mathematics*, 3(2):255–265, 1990.
- 38 Richard M. Karp. The probabilistic analysis of some combinatorial search algorithms. *Algorithms and complexity: New directions and recent results*, 1:19, 1976.
- 39 Adam Klivans and Rocco A. Servedio. Boosting and hard-core set construction. *Machine Learning*, 51(3):217–238, 2003.
- 40 Ilan Komargodski, Ran Raz, and Avishay Tal. Improved average-case lower bounds for demorgan formula size. In *Foundations of Computer Science (FOCS), 2013 IEEE 54th Annual Symposium on*, pages 588–597. IEEE, 2013.
- 41 Shachar Lovett and Srikanth Srinivasan. Correlation bounds for poly-size[AC<sup>0</sup>] circuits with  $n^{1-o(1)}$  symmetric gates. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 640–651. Springer, 2011.
- 42 A.A. Markov. On the inversion complexity of a system of functions. *Journal of the ACM (JACM)*, 5(4):331–334, 1958.
- 43 Ryan O’Donnell. Hardness amplification within NP. In *Proceedings of the thirty-fourth annual ACM symposium on Theory of computing*, pages 751–760. ACM, 2002.
- 44 Ryan O’Donnell and Karl Wimmer. KKL, Kruskal-Katona, and monotone nets. In *Foundations of Computer Science, 2009. FOCS’09. 50th Annual IEEE Symposium on*, pages 725–734. IEEE, 2009.
- 45 Aaron Potechin. Bounds on monotone switching networks for directed connectivity. In *Foundations of Computer Science (FOCS), 2010 51st Annual IEEE Symposium on*, pages 553–562. IEEE, 2010.
- 46 Ran Raz and Pierre McKenzie. Separation of the monotone NC hierarchy. In *Foundations of Computer Science, 1997. Proceedings., 38th Annual Symposium on*, pages 234–243. IEEE, 1997.



- 47 Ran Raz and Avi Wigderson. Probabilistic communication complexity of boolean relations. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 562–567. IEEE Comput. Soc. Press, 1989.
- 48 Ran Raz and Avi Wigderson. Monotone circuits for matching require linear depth. *Journal of the ACM (JACM)*, 39(3):736–744, 1992.
- 49 Alexander Razborov and Avi Wigderson.  $n^{\Omega(\log n)}$  lower bounds on the size of depth-3 threshold circuits with AND gates at the bottom. *Information Processing Letters*, 45(6):303–307, 1993.
- 50 Alexander A. Razborov. Lower bounds on monotone complexity of the logical permanent. *Mathematical Notes*, 37(6):485–493, 1985.
- 51 Alexander A. Razborov. Lower bounds on the monotone complexity of some boolean functions. *Doklady Akademii Nauk SSSR*, 281:798–801, 1985. English translation in *Soviet Math. Doklady* 31 (1985), 354–357.
- 52 Alexander A Razborov. On the method of approximations. In *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 167–176. ACM, 1989.
- 53 Alexander A. Razborov and Steven Rudich. Natural proofs. In *Proceedings of the twenty-sixth annual ACM symposium on Theory of computing*, pages 204–213. ACM, 1994.
- 54 Benjamin Rossman. On the constant-depth complexity of  $k$ -clique. In *Proceedings of the 40th annual ACM symposium on Theory of computing*, pages 721–730. ACM, 2008.
- 55 Benjamin Rossman. The monotone complexity of  $k$ -clique on random graphs. In *Foundations of Computer Science (FOCS), 2010 51st Annual IEEE Symposium on*, pages 193–201. IEEE, 2010.
- 56 Benjamin Rossman. Formulas vs. circuits for small distance connectivity. *arXiv preprint arXiv:1312.0355*, 2013.
- 57 Rocco A. Servedio. Monotone boolean formulas can approximate monotone linear threshold functions. *Discrete Applied Mathematics*, 142(1):181–187, 2004.
- 58 Alexander A. Sherstov. Communication complexity under product and nonproduct distributions. In *Computational Complexity, 2008. CCC'08. 23rd Annual IEEE Conference on*, pages 64–70. IEEE, 2008.
- 59 P.M. Spira. On time-hardware complexity tradeoffs for boolean functions. In *Proceedings of the 4th Hawaii Symposium on System Sciences*, pages 525–527, 1971.
- 60 Volker Strassen. The existence of probability measures with given marginals. *The Annals of Mathematical Statistics*, pages 423–439, 1965.
- 61 Éva Tardos. The gap between monotone and non-monotone circuit complexity is exponential. *Combinatorica*, 8(1):141–142, 1988.
- 62 Praseon Tiwari and Martin Tompa. A direct version of Shamir and Snir’s lower bounds on monotone circuit depth. *Information Processing Letters*, 49(5):243–248, 1994.
- 63 Leslie G. Valiant. Short monotone formulae for the majority function. *Journal of Algorithms*, 5(3):363–366, 1984.
- 64 Leslie G. Valiant. Negation is powerless for boolean slice functions. *SIAM Journal on Computing*, 15(2):531–535, 1986.
- 65 Emanuele Viola. Correlation bounds for polynomials over  $\{0,1\}$ . *ACM SIGACT News*, 40(1):27–44, 2009.
- 66 Ingo Wegener. Relating monotone formula size and monotone depth of boolean functions. *Information Processing Letters*, 16(1):41–42, 1983.
- 67 Ingo Wegener. More on the complexity of slice functions. *Theoretical computer science*, 43:201–211, 1986.

### A Proof of Lemma 3.4 (Persistent Minterms Under $\vee$ and $\wedge$ )

To simplify notation, we write  $f_s$  for  $f^{\vee \rho_s}$  and  $g_s$  for  $g^{\vee \rho_s}$ .

Proof of (3): Consider any  $x \in \mathcal{M}_d^{\vec{\rho}}(f \vee g)$ . Fix  $0 \leq s \leq t \leq m$  such that  $t - s \geq \langle \frac{d}{|x|} \rangle$  and  $x \in \mathcal{M}(f_s \vee g_s) \cap \mathcal{M}(f_t \vee g_t)$ . Since  $x$  is a minterm of  $f_s \vee g_s$ , we have  $f_s(x) = 1$  or  $g_s(x) = 1$ . Without loss of generality, assume  $f_s(x) = 1$ . We claim that  $x$  is also a minterm of  $f_t$ . Clearly  $f_t(x) = 1$  since  $f_s \leq f_t$ . It suffices to show that  $f_t(y) = 0$  for all  $y < x$ . This follows from the fact that  $x$  is a minterm of  $f_t \vee g_t$ , hence  $(f_t \vee g_t)(y) = 0$  for all  $y < x$ . Therefore,  $x \in \mathcal{M}(f_s) \cap \mathcal{M}(f_t)$ . Since  $t - s \geq \langle \frac{d}{j} \rangle \geq \langle \frac{d-1}{j} \rangle$ , we conclude that  $x \in \mathcal{M}_{d-1}^{\vec{\rho}}(f)$ .

Proof of (4): Consider any  $x \in \mathcal{M}_d^{\vec{\rho}}(f \wedge g)$ . Fix  $0 \leq s \leq t \leq m$  such that  $t - s \geq \langle \frac{d}{|x|} \rangle$  and  $x \in \mathcal{M}(f_s \wedge g_s) \cap \mathcal{M}(f_t \wedge g_t)$ . Let  $\ell := t - s$ . We will construct, by induction on  $i = 0, 1, \dots, \ell$ , two sequences  $y_0 \geq y_1 \geq \dots \geq y_\ell$  and  $z_0 \geq z_1 \geq \dots \geq z_\ell$  such that  $y_i \in \mathcal{M}(f_{s+i})$  and  $z_i \in \mathcal{M}(g_{s+i})$  and  $y_i \vee z_i = x$ :

- For the base case  $i = 0$ , since  $x$  is a minterm of  $f_s \wedge g_s$ , we have  $f_s(x) = g_s(x) = 1$ . Therefore, there exist  $y \in \mathcal{M}(f_s)$  and  $z \in \mathcal{M}(g_s)$  such that  $y, z \leq x$ . Note that  $(f_s \wedge g_s)(y \vee z) = 1$  and  $y \vee z \leq x$ . Again using the fact that  $x$  is a minterm of  $f_s \wedge g_s$ , it follows that  $y \vee z = x$ . These are the starting terms of our sequence:  $y_0 = y$  and  $z_0 = z$ .
- For the induction step, suppose we have chosen  $y_{i-1} \in \mathcal{M}(f_{s+i-1})$  and  $z_{i-1} \in \mathcal{M}(g_{s+i-1})$  such that  $y_{i-1} \vee z_{i-1} = x$ . Since  $f_{s+i-1} \leq f_{s+i}$  and  $g_{s+i-1} \leq g_{s+i}$ , we have  $f_{s+i}(y_{i-1}) = g_{s+i}(z_{i-1}) = 1$ . Therefore, there exist  $y \in \mathcal{M}(f_{s+i})$  and  $z \in \mathcal{M}(g_{s+i})$  such that  $y \leq y_{i-1}$  and  $z \leq z_{i-1}$ . Note that  $(f_{s+i} \wedge g_{s+i})(y \vee z) = 1$  and  $y \vee z \leq x$ . Since  $x$  is a minterm of  $f_{s+i} \wedge g_{s+i}$ , it follows that  $y \vee z = x$ . These are the next terms in our sequence:  $y_i = y$  and  $z_i = z$ .

Having constructed sequences  $\vec{y}, \vec{z} \in \text{Seq}_{\leq}^{\ell}(\{0, 1\}^n)$ , we finish the proof using Lemma 3.2. Since  $\ell \geq \langle \frac{d}{|x|} \rangle \geq \langle \frac{d}{|y_0|} \rangle$ , we may apply Lemma 3.2 to the reversed sequence  $(y_\ell, y_{\ell-1}, \dots, y_0) \in \text{Seq}_{\leq}^{\ell}(\{0, 1\}^n)$ ; we get  $0 \leq a \leq b \leq \ell$  such that  $y_a = y_b$  and  $b - a \geq \langle \frac{d-1}{|y_a|} \rangle$ . Therefore,  $y_a \in \mathcal{M}_{d-1}^{\vec{\rho}}(f)$ . Similarly, we get  $z_c \in \mathcal{M}_{d-1}^{\vec{\rho}}(g)$  for some  $0 \leq c \leq \ell$ . Since  $y_0 \leq y_a \leq y_\ell$  and  $z_0 \leq z_c \leq z_\ell$  and  $z_0 \vee y_0 = y_\ell \vee z_\ell = x$ , we conclude that  $y_a \vee z_c = x$ . ◀

### B Proof of Lemma 5.12 (Pathset Complexity and Formula Size)

Assume  $\Phi$  is a monotone formula and  $\vec{\rho} \in \text{Seq}_{\leq}^m(\mathcal{G})$  such that  $\mathcal{P}_A^{\vec{\rho}}(\Psi)$  is  $\varepsilon$ -small for every subformula  $\Psi$  of  $\Phi$  and every  $A \subseteq K$ .

Consider any  $\phi \in \text{Leaves}(\Phi)$  labeled by the indicator variable for a potential edge  $v^{(i)}w^{(j)}$ . Clearly  $\mathcal{P}_A^{\vec{\rho}}(\phi) = \emptyset$  for all  $A \subseteq K$  except possibly when  $E_A = \{vw\}$ , in which case the only possibility for  $\mathcal{P}_A^{\vec{\rho}}(\phi)$  other than  $\emptyset$  is the singleton pathset  $\{A'\}$  where  $A'$  is the  $A$ -section with  $E_{A'} = \{v^{(i)}w^{(j)}\}$ . It follows that  $\sum_{A \subseteq K} |\mathcal{P}_A^{\vec{\rho}}(\phi)| \leq 1$ .

Next, consider  $\Psi \in \text{Sub}(\Phi)$  with an  $\vee$ -gate on top:  $\Psi = \Psi_1 \vee \Psi_2$ . For all  $A \subseteq K$ , by Lemma 5.11, we have  $\mathcal{P}_A^{\vec{\rho}}(\Psi) \subseteq \mathcal{P}_A^{\vec{\rho}}(\Psi_1) \cup \mathcal{P}_A^{\vec{\rho}}(\Psi_2)$ . By properties (monotonicity) and (sub-additivity) of  $\chi_\varepsilon$ , it follows that

$$\chi_\varepsilon(\mathcal{P}_A^{\vec{\rho}}(\Psi)) \leq \chi_\varepsilon(\mathcal{P}_A^{\vec{\rho}}(\Psi_1)) + \chi_\varepsilon(\mathcal{P}_A^{\vec{\rho}}(\Psi_2)). \quad (23)$$

Now consider  $\Psi = \Psi_1 \wedge \Psi_2 \in \text{Sub}(\Phi)$ . By Lemma 5.11, we have

$$\mathcal{P}_A^{\vec{\rho}}(\Psi) \subseteq \mathcal{P}_A^{\vec{\rho}}(\Psi_1) \cup \mathcal{P}_A^{\vec{\rho}}(\Psi_2) \cup \bigcup_{B, C \subseteq A: B \cup C = A} \mathcal{P}_B^{\vec{\rho}}(\Psi_1) \bowtie \mathcal{P}_C^{\vec{\rho}}(\Psi_2). \quad (24)$$

(This expression extracts from (16) the case where  $B = A$ , noting that  $\mathcal{P}_A^{\vec{\rho}}(\Psi_1) \boxtimes \mathcal{P}_C^{\vec{\rho}}(\Psi_2) \subseteq \mathcal{P}_A^{\vec{\rho}}(\Psi_1)$ ; and similarly the case where  $C = A$ .) By properties (monotonicity), (sub-additivity) and (join inequality) of  $\chi_\varepsilon$ ,

$$\begin{aligned} \chi_\varepsilon(\mathcal{P}_A^{\vec{\rho}}(\Psi)) &\leq \chi_\varepsilon(\mathcal{P}_A^{\vec{\rho}}(\Psi_1)) + \chi_\varepsilon(\mathcal{P}_A^{\vec{\rho}}(\Psi_2)) + \sum_{B, C \not\subseteq A: B \cup C = A} (\chi_\varepsilon(\mathcal{P}_B^{\vec{\rho}}(\Psi_1)) + \chi_\varepsilon(\mathcal{P}_C^{\vec{\rho}}(\Psi_2))) \\ &\leq \left( \chi_\varepsilon(\mathcal{P}_A^{\vec{\rho}}(\Psi_1)) + 2^k \sum_{B \not\subseteq A} \chi_\varepsilon(\mathcal{P}_B^{\vec{\rho}}(\Psi_1)) \right) + \left( \chi_\varepsilon(\mathcal{P}_A^{\vec{\rho}}(\Psi_2)) + 2^k \sum_{B \not\subseteq A} \chi_\varepsilon(\mathcal{P}_B^{\vec{\rho}}(\Psi_2)) \right). \end{aligned} \quad (25)$$

If we now start with  $\chi_\varepsilon(\mathcal{P}_K^{\vec{\rho}}(\Phi))$  and repeatedly expand according to (25) and (23) down to the leaves of  $\Phi$ , we get a bound of the form

$$\mathcal{P}_K^{\vec{\rho}}(\Phi) \leq \sum_{\phi \in \text{Leaves}(\Phi)} \sum_{A \subseteq K} c_{\phi, A} \cdot \chi_\varepsilon(\mathcal{P}_A^{\vec{\rho}}(\phi))$$

for some  $c_{\phi, A} \in \mathbb{N}$ . For  $\phi \in \text{Leaves}(\Phi)$  at depth  $d$  ( $\leq \text{depth}(\Phi)$ ), the coefficient  $c_{\phi, A}$  equals the sum, over all chains  $K = B_0 \supset B_1 \supset \dots \supset B_t = A$ , of  $2^{kt}$  times the binomial coefficient  $\binom{d}{t}$  (counting the locations of the  $\wedge$ -gates above  $\phi$  where branching occurred in the expansion of (25)). Thus, we have the upper bound  $c_{\phi, A} \leq 2^{O(k^2)} \cdot \text{depth}(\Phi)^k$ . Using the fact that  $\sum_{A \subseteq K} |\mathcal{P}_A^{\vec{\rho}}(\phi)| \leq 1$  for all  $\phi \in \text{Leaves}(\Phi)$  and the definition  $\text{size}(\Phi) = |\text{Leaves}(\Phi)|$ , we conclude that  $\mathcal{P}_K^{\vec{\rho}}(\Phi) \leq 2^{O(k^2)} \cdot \text{depth}(\Phi)^k \cdot \text{size}(\Phi)$ . ◀

## C Proof of Lemma 1.3 (Negation-Limited Circuits)

Our proof of Lemma 1.3 combines a monotone coupling theorem of Holley [30] (which is the main ingredient in the proof of his generalization the FKG inequalities [22]) with an observation about negations in circuits due to Amano and Maruoka [7]. We require one definition:

► **Definition 3.1.** For a boolean (not necessarily monotone) function  $h : \{0, 1\}^n \rightarrow \{0, 1\}$ , let

$$\text{mon-pairs}(h) := \{(x, y) \in \{0, 1\}^n \times \{0, 1\}^n : h(x) = 0 \text{ and } h(y) = 1 \text{ and } x < y\}.$$

The following lemma and its proof are adapted from Theorem 3.2 of [7]. The only difference is that we consider all monotone pairs, rather than only the monotone boundary (i.e. only monotone pairs  $(x, y)$  with  $|y| - |x| = 1$ ).

► **Lemma 3.2.** *For every circuit  $\mathcal{C}$  with  $t$  negation gates, there exist  $t' = 2^{t+1} - 1$  monotone circuits  $\mathfrak{M}_1, \dots, \mathfrak{M}_{t'}$  of the same size and depth such that  $\text{mon-pairs}(\mathcal{C}) \subseteq \bigcup_{i=1}^{t'} \text{mon-pairs}(\mathfrak{M}_i)$ .*

**Proof.** Let  $\mathcal{C}_1, \dots, \mathcal{C}_t$  be the sub-circuits of  $\mathcal{C}$  which feed directly into negation gates, listed in “topological order” such that  $i < j$  whenever  $\mathcal{C}_i$  is a sub-circuit of  $\mathcal{C}_j$ . Also, let  $\mathcal{C}_{t+1}$  be  $\mathcal{C}$  itself. For every  $j \in \{1, \dots, t+1\}$  and  $\alpha \in \{0, 1\}^{j-1}$ , let  $\mathfrak{M}_\alpha$  be the monotone circuit obtained from  $\mathcal{C}_j$  by, for each  $i \in \{1, \dots, j-1\}$  such that  $\mathcal{C}_i$  is a sub-circuit of  $\mathcal{C}_j$ , replacing the negation gate above  $\mathcal{C}_i$  with the constant  $\alpha_i$ . The number of these monotone circuits is  $\sum_{j=1}^{t+1} 2^{j-1} = 2^{t+1} - 1$ . To finish the argument, consider any  $(x, y) \in \text{mon-pairs}(\mathcal{C})$ . Let  $j$  be the first index such that  $\mathcal{C}_j(x) \neq \mathcal{C}_j(y)$ , and let  $\alpha \in \{0, 1\}^{j-1}$  be the element  $\alpha_i := \mathcal{C}_i(x) = \mathcal{C}_i(y)$ . Then  $(x, y) \in \text{mon-pairs}(\mathfrak{M}_\alpha)$ . We conclude that  $\text{mon-pairs}(\mathcal{C}) \subseteq \bigcup_{j \in [t+1]} \bigcup_{\alpha \in \{0, 1\}^{j-1}} \text{mon-pairs}(\mathfrak{M}_\alpha)$ . ◀

► **Lemma 3.3** (Holley [30]). *Let  $\mu_0, \mu_1$  be two strictly positive probability distributions on  $\{0, 1\}^n$  which satisfy the “Holley condition”*

$$\mu_0(x)\mu_1(y) \leq \mu_0(x \wedge y)\mu_1(x \vee y) \quad \text{for all } x, y. \quad (26)$$

*Then there exists a probability distribution  $\nu$  on  $\{0, 1\}^n \times \{0, 1\}^n$  (“monotone coupling of  $\mu_0$  and  $\mu_1$ ”) such that*

- $\sum_y \nu(x, y) = \mu_0(x)$  for all  $x$ ,
- $\sum_x \nu(x, y) = \mu_1(y)$  for all  $y$ ,
- $\nu(x, y) > 0 \Rightarrow x \leq y$  for all  $x, y$ .

Holley’s proof of Lemma 3.3 uses a Markov chain coupling argument. We remark that Lemma 3.3 also follows from an earlier (and much more general) monotone coupling theorem of Strassen [60].

► **Lemma 3.4.** *Let  $\mu$  be a distribution on  $\{0, 1\}^n$  which satisfies the FKG lattice condition (1), and let  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  be a monotone function such that  $\mathbb{E}_\mu(f) \in (0, 1)$ . For  $b \in \{0, 1\}$ , define the distribution  $\mu_b$  on  $\{0, 1\}^n$  by*

$$\mu_b(x) := \begin{cases} \mu(x)/(1 - \mathbb{E}_\mu(f)) & \text{if } f(x) = b = 0, \\ \mu(x)/\mathbb{E}_\mu(f) & \text{if } f(x) = b = 1, \\ 0 & \text{otherwise.} \end{cases} \quad (27)$$

*Then the pair  $\mu_0, \mu_1$  satisfy the Holley condition (26).*

**Proof.** We simply observe:

- If  $f(x) = 1$ , then  $\mu_0(x) = \mu_0(x \wedge y) = 0$ .
- If  $f(y) = 0$ , then  $\mu_1(y) = \mu_1(x \vee y) = 0$ .
- If  $f(x) = 0$  and  $f(y) = 1$ , then

$$\mu_0(x)\mu_1(y) = \frac{\mu(x)\mu(y)}{\mathbb{E}_\mu(f)(1 - \mathbb{E}_\mu(f))} \leq \frac{\mu(x \wedge y)\mu(x \vee y)}{\mathbb{E}_\mu(f)(1 - \mathbb{E}_\mu(f))} = \mu_0(x \wedge y)\mu_1(x \vee y). \quad \blacktriangleleft$$

**Proof of Lemma 1.3.** Let  $\mu$  be a distribution on  $\{0, 1\}^n$  which satisfies the FKG lattice condition (1), and suppose  $f \in \mathbb{B}_n^+$  such that  $\mathbb{E}_\mu(f) = 1/2$  (i.e.  $f$  is balanced with respect to  $\mu$ ). We prove the contrapositive statement to Lemma 1.3. Assume  $\mathfrak{C}$  is a monotone circuit which computes  $f$  on  $\mu$  with advantage  $\delta$ , that is,

$$\mathbb{P}_{x \sim \mu} [\mathfrak{C}(x) = f(x)] = \frac{1}{2} + \delta.$$

We will show that  $f$  is computed with advantage  $\geq \delta/(2^{t+1} - 1)$  by a monotone circuit of the same size and depth.

Define  $\mu_0, \mu_1$  by (27) as in Lemma 3.4. By Lemma 3.3 there is a monotone coupling  $\nu$  of  $\mu_0, \mu_1$ , which is supported on mon-pairs( $f$ ). For every monotone function  $h \in \mathbb{B}_n^+$ , we have

$$\begin{aligned} \nu(\text{mon-pairs}(h)) &= \mathbb{E}_{(x,y) \sim \nu} [h(y) - h(x)] & (28) \\ &= \mathbb{E}_{(x,y) \sim \nu} [h(y)] - \mathbb{E}_{(x,y) \sim \nu} [h(x)] \\ &= \mathbb{P}_{(x,y) \sim \nu} [h(y) = 1] + \mathbb{P}_{(x,y) \sim \nu} [h(x) = 0] - 1 \\ &= 2 \left( \mathbb{P}_{y \sim \mu} [h(y) = f(y) = 1] + \mathbb{P}_{x \sim \mu} [h(x) = f(x) = 0] \right) - 1 \\ &= 2 \mathbb{P}_{x \sim \mu} [h(x) = f(x)] - 1. \end{aligned}$$

It follows from Lemma 3.2 that there exists a monotone circuit  $\mathfrak{M}$ , of the same size and depth as  $\mathfrak{C}$ , such that

$$\nu(\text{mon-pairs}(\mathfrak{M})) \geq \frac{1}{2^{t+1} - 1} \nu(\text{mon-pairs}(\mathfrak{C})).$$

We complete the proof by two applications of (28):

$$\begin{aligned} \mathbb{P}_{x \sim \mu} [\mathfrak{M}(x) = f(x)] &= \frac{1}{2} \left( 1 + \nu(\text{mon-pairs}(\mathfrak{M})) \right) \\ &\geq \frac{1}{2} \left( 1 + \frac{1}{2^{t+1} - 1} \nu(\text{mon-pairs}(\mathfrak{C})) \right) \\ &= \frac{1}{2} \left( 1 + \frac{1}{2^{t+1} - 1} \left( 2 \mathbb{P}_{x \sim \mu} [\mathfrak{C}(x) = f(x)] - 1 \right) \right) \\ &= \frac{1}{2} + \frac{\delta}{2^{t+1} - 1}. \end{aligned}$$

◀

# Non-Commutative Formulas and Frege Lower Bounds: a New Characterization of Propositional Proofs

Fu Li<sup>1</sup>, Iddo Tzameret<sup>2</sup>, and Zhengyu Wang<sup>3</sup>

1 Institute for Interdisciplinary Information Sciences, Tsinghua University\*

2 Department of Computer Science, Royal Holloway, University of London  
Iddo.Tzameret@rhul.ac.uk

3 Department of Computer Science, Harvard University

---

## Abstract

---

Does every Boolean tautology have a short propositional-calculus proof? Here, a propositional-calculus (i.e. Frege) proof is any proof starting from a set of axioms and deriving new Boolean formulas using a fixed set of sound derivation rules. Establishing any super-polynomial size lower bound on Frege proofs (in terms of the size of the formula proved) is a major open problem in proof complexity, and among a handful of fundamental hardness questions in complexity theory by and large. Non-commutative arithmetic formulas, on the other hand, constitute a quite weak computational model, for which exponential-size lower bounds were shown already back in 1991 by Nisan [20], using a particularly transparent argument.

In this work we show that Frege lower bounds in fact follow from corresponding size lower bounds on non-commutative formulas computing certain polynomials (and that such lower bounds on non-commutative formulas must exist, unless  $NP=coNP$ ). More precisely, we demonstrate a natural association between tautologies  $T$  to non-commutative polynomials  $p$ , such that:

- if  $T$  has a polynomial-size Frege proof then  $p$  has a polynomial-size non-commutative arithmetic formula; and conversely, when  $T$  is a DNF, if  $p$  has a polynomial-size non-commutative arithmetic formula over  $GF(2)$  then  $T$  has a Frege proof of quasi-polynomial size.

The argument is a characterization of Frege proofs as non-commutative formulas: we show that the Frege system is (quasi-) polynomially equivalent to a *non-commutative Ideal Proof System* (IPS), following the recent work of Grochow and Pitassi [10] that introduced a propositional proof system in which proofs are arithmetic circuits, and the work in [35] that considered adding the commutator as an axiom in algebraic propositional proof systems. This gives a characterization of propositional Frege proofs in terms of (non-commutative) arithmetic formulas that is tighter than (the formula version of IPS) in Grochow and Pitassi [10], in the following sense:

- (i) The non-commutative IPS is *polynomial-time checkable* – whereas the original IPS was checkable in probabilistic polynomial-time; and
- (ii) Frege proofs *unconditionally* quasi-polynomially simulate the non-commutative IPS – whereas Frege was shown to efficiently simulate IPS only assuming that the decidability of PIT for (commutative) arithmetic formulas by polynomial-size circuits is efficiently provable in Frege.


**1998 ACM Subject Classification** F.2.2 [Analysis of Algorithms and Problem Complexity] Non-numerical Algorithms and Problems – Complexity of proof procedures

**Keywords and phrases** Proof complexity, algebraic complexity, non-commutative formulas, Frege proofs, ideal proof system


**Digital Object Identifier** 10.4230/LIPIcs.CCC.2015.412

---

\* The first author was supported in part by the National Basic Research Program of China Grant 2011CBA00300, 2011CBA00301, the National Natural Science Foundation of China Grant 61033001, 61361136003, 61373002.

 © Fu Li, Iddo Tzameret, and Zhengyu Wang;  
licensed under Creative Commons License CC-BY  
30th Conference on Computational Complexity (CCC'15).

Editor: David Zuckerman; pp. 412–432

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

## 1 Introduction

### 1.1 Propositional proof complexity

The field of propositional proof complexity aims to understand and analyze the computational resources required to prove propositional statements. The problems the field poses are fundamental, difficult and go back to the work of Cook and Reckhow [8], who showed the immediate relevance of these problems to the  $NP$  vs.  $coNP$  problem (and thus the  $P$  vs.  $NP$  problem).

Among the major unsolved questions in proof complexity, is whether the standard propositional logic calculus, either in the form of the Sequent Calculus, or equivalently, in the axiomatic form of Hilbert proofs (i.e., Frege proofs), is polynomially bounded; that is, whether every propositional tautology (or unsatisfiable formula) has a proof whose size is polynomially bounded (refutation, resp.) in the size of the formula proved. Here, we consider the size of proofs as the number of symbols it takes to write them down, where each formula in the proof is written as a Boolean *formula* (in other words we count the total number of logical gates appearing in the proof where each proof-line is a formula). It is known [29] that all Frege proof-systems (formally, a Frege proof system is any propositional proof system with a fixed number of axiom schemes and sound derivation rules that is also implicational complete, and in which proof-lines are written as propositional formulas (see e.g., [14] and Definition 2.4 below)) as well as the Gentzen sequent calculus (with the cut rule) are polynomially equivalent to each other, and hence it does not matter precisely which rules, axioms, and logical-connectives we use.

Complexity-wise, the Frege proof system is considered a very strong system alas a poorly understood one. The qualification *strong* here has several meanings: first, that no super-polynomial lower bound is known for Frege proofs. Second, that there are not even good hard candidates for the Frege system (see [4, 17, 18] for a further discussion on hard proof complexity candidates). Third, that for most hard instances (e.g., the pigeonhole principle and Tseitin tautologies) that are known to be hard for weaker systems (e.g., resolution, cutting planes, etc.), there *are* known polynomial bounds on Frege proofs. Fourth, that proving super-polynomial lower bounds on Frege proofs seems to a certain extent out of reach of current techniques. And finally, that by the common (mainly informal) correspondence between circuits and proofs – namely, the correspondence between a circuit-class  $\mathcal{C}$  and a proof system in which every proof-line is written as a circuit from  $\mathcal{C}$  (to be more precise, one has to associate a circuit class  $\mathcal{C}$  with a proof system in which a *family* of proofs is written such that every proof-line in the family is a circuit family from  $\mathcal{C}$ ) – Frege system corresponds to the circuit class of polynomial-size  $\log(n)$ -depth circuits denoted  $NC^1$  (equivalently, of polynomial-size formulas [32]), considered to be a strong computational model for which no (explicit) super-polynomial lower bounds are currently known.

Accordingly, proving lower bounds on Frege proofs is considered an extremely hard task. In fact, the best lower bound known today is only quadratic [14], which uses a fairly simple syntactic argument. If we put further impeding restrictions on Frege proofs, like restricting the depth of each formula appearing in a proof to a certain fixed constant, exponential lower bounds can be obtained [1, 21, 21]. Although these constant-depth Frege exponential-size lower bounds go back to Ajtai's result from 1988, they are still in some sense the state-of-the-art in proof complexity lower bounds (beyond the important developments on weaker proof systems, such as resolution and its weak extensions). Constant-depth Frege lower bounds use quite involved probabilistic arguments, mainly specialized switching lemmas tailored for specific tautologies (namely, counting tautologies, most notable of which are the Pigeonhole



Principle tautologies). Even random  $k$ -CNF formulas near the satisfiability threshold are not known to be hard for constant-depth Frege (let alone hard for [unrestricted depth] Frege).

All of the above goes to emphasize the importance, basic nature and difficulty in understanding the complexity of strong propositional proof systems, while showing how little is actually known about these systems.

## 1.2 Prominent directions for understanding propositional proofs

As we already mentioned, there is a guiding line in proof complexity which states a correspondence between the complexity of circuits and the complexity of proofs. This correspondence is mainly informal, but there are seemingly good indications showing it might be more than a superficial analogy. One of the most compelling evidence for this correspondence is that there is a precise formal correspondence (cf. [7]) between the first-order logical theories of bounded arithmetic (whose axioms state the existence of sets taken from a given complexity class  $\mathcal{C}$ ) to propositional proof systems (in which proof-lines are circuits from  $\mathcal{C}$ ).

Another facet of the informal correspondence between circuit complexity and proof complexity is that circuit hardness can sometimes be used to obtain proof complexity hardness. The most notable example of this are the lower bounds on constant-depth Frege proofs mentioned above: constant-depth Frege proofs can be viewed as propositional logic operating with  $\mathbf{AC}^0$  circuits, and the known lower bounds on constant depth Frege proofs (cf. [1, 16, 21]) use techniques borrowed from  $\mathbf{AC}^0$  circuits lower bounds. The success in moving from circuit hardness towards proof-complexity hardness has spurred a flow of attempts to obtain lower bounds on proof systems other than constant depth Frege. For example, Pudlák [22] and Atserias et al. [2] studied proofs based on monotone circuits, motivated by known exponential lower bounds on monotone circuits. Raz and Tzameret [28, 27, 34] investigated algebraic proof systems operating with multilinear formulas, motivated by lower bounds on multilinear formulas for the determinant, permanent and other explicit polynomials [24, 23]. Atserias et al. [3], Krajíček [15] and Segerlind [31] have considered proofs operating with ordered binary decision diagrams (OBDDs), and the second author [35] initiated the study of proofs operating with non-commutative formulas (see Sec. 1.5 for a comparison with the current work).

Until quite recently it was unknown whether the correspondence between proofs and circuits is two-sided, namely, whether proof complexity hardness (of concrete known proof systems) can imply any computational hardness. An initial example of such an implication from proof hardness to circuit hardness was given by Raz and Tzameret [28]. They showed that a separation between algebraic proof systems operating with arithmetic circuits and multilinear arithmetic circuits, resp., for an explicit family of polynomials, implies a separation between arithmetic circuits and multilinear arithmetic circuits. In a recent significant development about the complexity of strong proof systems, Grochow and Pitassi [10] demonstrated a much stronger correspondence. They introduced a natural propositional proof system, called the *Ideal Proof System* (IPS for short), for which *any* super-polynomial size lower bound on IPS implies a corresponding size lower bound on arithmetic circuits, and formally, that the permanent does not have polynomial-size arithmetic circuits. The IPS is defined as follows:

► **Definition 1.1** (Ideal Proof System (IPS) [10]). Let  $F_1(\bar{x}), \dots, F_m(\bar{x})$  be a system of polynomials in the variables  $x_1, \dots, x_n$ , where the polynomials  $x_i^2 - x_i$ , for all  $1 \leq i \leq n$ , are part of this system. An *IPS refutation (or certificate)* that the  $F_i$ 's polynomials have no 0-1 solutions is a polynomial  $C(\bar{x}, \bar{y})$  in the variables  $x_1, \dots, x_n$  and  $y_1, \dots, y_m$ , such that:

1.  $F(x_1, \dots, x_n, \bar{0}) = 0$ ; and
2.  $F(x_1, \dots, x_n, F_1(\bar{x}), \dots, F_m(\bar{x})) = 1$ .



The essence of IPS is that a proof (or refutation) is a *single* polynomial that can be written simply as an arithmetic *circuit* or *formula*. The advantage of this formulation is that now we can obtain direct connections between circuit/formula hardness (i.e., “computational hardness”) and hardness of proofs. Grochow and Pitassi showed indeed that a lower bound on IPS written as an arithmetic circuit implies that the permanent does not have polynomial-size algebraic circuits (Valiant’s conjectured separation  $\text{VNP} \neq \text{VP}$  [36]); And similarly, a lower bound on IPS written as an arithmetic *formula* implies that the permanent does not have polynomial-size algebraic formulas ( $\text{VNP} \neq \text{VP}_e$ , *ibid*).

Under certain assumptions, Grochow and Pitassi [10] were able to connect their result to standard propositional-calculus proof systems, i.e., Frege and Extended Frege. Their assumption was the following: *Frege has polynomial-size proofs of the statement expressing that the PIT for arithmetic formulas is decidable by polynomial-size Boolean circuits* (PIT for arithmetic formulas is the problem to decide whether an input arithmetic formula computes the (formal) zero polynomial). They showed that under this assumption super-polynomial lower bounds on Frege proofs imply that the permanent does not have polynomial-size arithmetic circuits. This, in turn, can be considered as a (conditional) justification for the apparent difficulty in proving lower bounds on strong proof systems (We focus only on the relevant results about Frege proofs from [10]; and not the results about Extended Frege in [10]; the latter proof system operates, essentially, with Boolean circuits, in the same way that Frege operates with Boolean formulas (equivalently  $\text{NC}^1$  circuits)).

### 1.3 Overview of results and proofs

#### 1.3.1 Sketch

In this paper we contribute to the understanding of strong proof systems such as Frege, and to the fundamental search for lower bounds on these systems, by formulating a very natural proof system – a non-commutative variant of the ideal proof system – which we show captures *unconditionally* (up to a quasi-polynomial-size increase) propositional Frege proofs. A proof in the non-commutative IPS is simply a *single non-commutative polynomial* written as a non-commutative formula. This gives a fairly compelling and simple new characterization of the proof complexity of propositional Frege proofs. Moreover, it brings new hope for achieving lower bounds on strong proof systems, by reducing the task of lower bounding Frege proofs to the following seemingly much more manageable task: proving matrix rank lower bounds on the matrices associated with certain non-commutative polynomials (in the sense of Nisan [20]; see below for details).

We also tighten the results in Grochow and Pitassi [10], in the sense that we show that in order to obtain a characterization of Frege proofs in terms of an ideal proof system it is advantageous to consider non-commutative polynomials instead of commutative ones (as well as to add the commutator axioms). This shows that, at least for Frege, and in the framework of the ideal proof system, lower bounds on Frege proofs do not necessarily entail in themselves very strong computational lower bounds.

#### 1.3.2 Some preliminaries: non-commutative polynomials and formulas

A *non-commutative polynomial* over a given field  $\mathbb{F}$  and with the variables  $X := \{x_1, x_2, \dots\}$  is a formal sum of monomials with coefficients from  $\mathbb{F}$  such that the product of variables is non-commuting. For example,  $x_1x_2 - x_2x_1 + x_3x_2x_3^2 - x_2x_3^3$ ,  $x_1x_2 - x_2x_1$  and 0 are three distinct polynomials in  $\mathbb{F}\langle X \rangle$ . The ring of non-commutative polynomials with variables  $X$  with coefficients from  $\mathbb{F}$  is denoted  $\mathbb{F}\langle X \rangle$ .

A *polynomial* (i.e., a *commutative* polynomial) over a field is defined in the same way as a non-commutative polynomial except that now the product of variables *is* commutative; that is, it is a sum of (commutative) monomials.

A *non-commutative arithmetic formula* (non-commutative formula for short; see Definition 2.5) is a fan-in two labeled tree, with edges directed from leaves towards the root, such that the leaves are labeled with field elements (for a given field  $\mathbb{F}$ ) or variables  $x_1, \dots, x_n$ , and internal nodes (including the root) are labeled with a plus  $+$  or product  $\times$  gates. A product gate has an order on its two children (holding the order of non-commutative product). A non-commutative formula computes a non-commutative polynomial in the natural way (see Definition 2.5).

Exponential-size lower bounds on non-commutative formulas (over any field) were established by Nisan [20]. The idea (in retrospect) is quite simple: first transform a non-commutative formula into an algebraic branching program (ABP); and then show that the number of nodes in the  $i$ th layer of an ABP computing a degree  $d$  homogenous non-commutative polynomial  $f$  is bounded from below by the rank of the degree  $i$ -partial-derivative matrix of  $f$ . (The degree  $i$  partial derivative matrix of  $f$  is the matrix whose rows are all non-commutative monomials of degree  $i$  and columns are all non-commutative monomials of degree  $d - i$ , such that the entry in row  $M$  and column  $N$  is the coefficient of the  $d$  degree monomial  $M \cdot N$  in  $f$ .) Thus, lower bounds on non-commutative formulas follow from quite immediate rank arguments (e.g., the partial derivative matrices associated with the permanent and determinant can easily be shown to have high ranks).

### 1.3.3 Non-commutative ideal proof system

Recall the IPS refutation system in Definition 1.1 above. We use the idea introduced in [35], that considered adding the commutator  $x_1x_2 - x_2x_1$  as an axiom in propositional algebraic proof systems, to define a refutation system that polynomially simulates Frege:

► **Definition 1.2** (Non-commutative IPS). Let  $\mathbb{F}$  be a field. Assume that  $F_1(\bar{x}) = F_2(\bar{x}) = \dots = F_m(\bar{x}) = 0$  is a system of non-commutative polynomial equations from  $\mathbb{F}\langle x_1, \dots, x_n \rangle$ , and suppose that the following set of equations (axioms) are included in the  $F_i(\bar{x})$ 's:

**Boolean axioms:**  $x_i \cdot (1 - x_i)$ , for all  $1 \leq i \leq n$ ;

**Commutator axioms:**  $x_i \cdot x_j - x_j \cdot x_i$ , for all  $1 \leq i < j \leq n$ .

Suppose that the  $F_i(\bar{x})$ 's have no common 0-1 solutions. (One can check that the  $F_i(\bar{x})$ 's have no common 0-1 solutions in  $\mathbb{F}$  iff they do not have a common 0-1 solution in every  $\mathbb{F}$ -algebra.) A *non-commutative IPS refutation* (or *certificate*) that the system of  $F_i(\bar{x})$ 's is unsatisfiable is a non-commutative polynomial  $\mathfrak{F}(\bar{x}, \bar{y})$  in the variables  $x_1, \dots, x_n$  and  $y_1, \dots, y_m$  (i.e.  $\mathfrak{F} \in \mathbb{F}\langle \bar{x}, \bar{y} \rangle$ ), such that:

1.  $\mathfrak{F}(x_1, \dots, x_n, \bar{0}) = 0$ ; and
2.  $\mathfrak{F}(x_1, \dots, x_n, F_1(\bar{x}), \dots, F_m(\bar{x})) = 1$ .

We always assume that the non-commutative IPS refutation is written as a *non-commutative formula*. Hence the *size* of a non-commutative IPS refutation is the minimal size of a non-commutative formula computing the non-commutative IPS refutation.

The main result of this paper is that the non-commutative IPS (over either  $\mathbb{Q}$  or  $\mathbb{Z}_q$ , for any prime  $q$ ) polynomially simulates Frege; and conversely, Frege quasi-polynomially simulates the non-commutative IPS (over  $GF(2)$ ). We explain the results in what follows.

For the purpose of the next theorem, we use a standard translation of propositional formulas  $T$  into non-commutative arithmetic formulas:

► **Definition 1.3.** Let  $\text{tr}(x_i) := x_i$ , for variables  $x_i$ ;  $\text{tr}(\text{false}) := 1$ ;  $\text{tr}(\text{true}) := 0$ ; and by induction on the size of the propositional formula:  $\text{tr}(\neg T_1) := 1 - \text{tr}(T_1)$ ;  $\text{tr}(T_1 \vee T_2) = \text{tr}(T_1) \cdot \text{tr}(T_2)$  and finally  $\text{tr}(T_1 \wedge T_2) = 1 - ((1 - \text{tr}(T_1)) \cdot (1 - \text{tr}(T_2)))$ .

For a non-commutative formula  $f$  denote by  $\widehat{f}$  the non-commutative polynomial computed by  $f$ . Thus,  $T$  is a propositional tautology (i.e., a Boolean formula that is satisfied by every assignment) iff  $\widehat{\text{tr}(T)} = 0$  for every 0-1 assignment to the underlying variables of the non-commutative polynomial.

► **Theorem 1.4.** Let  $\mathbb{F}$  be either  $\mathbb{Q}$  or  $\mathbb{Z}_q$ , for a prime  $q$ . The non-commutative IPS refutation system, when refutations are written as non-commutative formulas over  $\mathbb{F}$ , polynomially simulates the Frege system. More precisely, for every propositional tautology  $T$ , if  $T$  has a polynomial-size Frege proof then there is a non-commutative IPS certificate (over  $\mathbb{F}$ ) of  $\text{tr}(\neg T)$  that has a polynomial non-commutative formula size.

The proof of Theorem 1.4 proceeds as follows. To simulate Frege proofs we use an intermediate proof system  $\mathcal{F}\text{-}\mathcal{PC}$  formulated by Grigoriev and Hirsch [9]. The  $\mathcal{F}\text{-}\mathcal{PC}$  system (Definition 2.7) is akin to the polynomial calculus refutation system [6], except that we operate in  $\mathcal{F}\text{-}\mathcal{PC}$  with arithmetic formulas treated as syntactic terms, instead of writing polynomials throughout the proof as sum of monomials. We have the two rules of polynomial calculus: from a pair of previously derived polynomials  $f, g$  we can derive  $af + bg$  for  $a, b \in \mathbb{F}$ , and from  $f$  we can derive  $x_i \cdot f$ , for any variable  $x_i$ . We also have local rewriting rules, that can operate on any sub-formula of an arithmetic formula appearing in the proof. These rewriting rules express simple operations on polynomials like commutativity of addition and product, associativity, distributivity, etc.

Grigoriev and Hirsch [9] showed that  $\mathcal{F}\text{-}\mathcal{PC}$  polynomially simulates Frege proofs, and that for tree-like Frege proofs the polynomial simulation yields tree-like  $\mathcal{F}\text{-}\mathcal{PC}$  proofs. Since tree-like Frege is polynomially equivalent to Frege (because Frege proofs can always be balanced to depth which is logarithmic in their size; cf. [14] for a proof), we have that tree-like  $\mathcal{F}\text{-}\mathcal{PC}$  polynomially simulates (dag-like) Frege proofs.

To conclude Theorem 1.4 it therefore remains to prove that non-commutative IPS polynomially simulates tree-like  $\mathcal{F}\text{-}\mathcal{PC}$  proofs. This can be proved by induction on the number of lines in the  $\mathcal{F}\text{-}\mathcal{PC}$  proofs. The interesting case in the induction is the simulation of the commutativity rewrite-rule for products by the non-commutative IPS system, which is done using the commutator axioms.

Now, since we write refutations as non-commutative formulas we can use the polynomial-time deterministic Polynomial Identity Testing algorithm for non-commutative formulas, devised by Raz and Shpilka [26], to check in deterministic polynomial-time the correctness of non-commutative IPS refutations. Therefore, we obtain:

► **Corollary 1.5.** The non-commutative IPS is a sound and complete Cook-Reckhow refutation system. That is, it is a sound and complete refutation system for unsatisfiable propositional formulas in which refutations can be checked for correctness in deterministic polynomial-time.

This should be contrasted with the original (commutative) IPS of [10], for which verification of refutations is done in probabilistic polynomial time (using the standard Schwartz-Zippel [30, 37] PIT algorithm).

The major consequence of Theorem 1.4 is that to prove a super-polynomial Frege lower bound it is now sufficient to prove a super-polynomial lower bound on non-commutative

formulas computing certain polynomials. Specifically, it is enough to prove that any non-commutative IPS certificate  $\mathfrak{F}(\bar{x}, \bar{y})$  (which is simply a non-commutative polynomial) has a super-polynomial non-commutative formula size; and yet in another words, it suffices to show that any such  $\mathfrak{F}$  must have a super-polynomial total rank according to the associated partial-derivatives matrices discussed before.

We now consider the *other direction*, namely, whether Frege can simulate the non-commutative IPS. We show that it does for CNFs (this is the case considered in [10]), over  $GF(2)$ , and with only a quasi-polynomial increase in size. For convenience, we use a slightly different translation of clauses to non-commutative formulas than Definition 1.3:

► **Definition 1.6.** Given a Boolean formula  $f$  we define the non-commutative formula translation  $\text{tr}'(f)$  as follows. Let  $\text{tr}'(x) := 1 - x$  and  $\text{tr}'(\neg x) := x$ , for  $x$  a variable. And let  $\text{tr}'(f_1 \vee \dots \vee f_r) := \text{tr}'(f_1) \cdots \text{tr}'(f_r)$  (where the sequence of products stands for a tree of product gates with  $\text{tr}'(f_i)$  as leaves). Further, for a clause  $k_i$  in a CNF  $\phi = k_1 \wedge k_2 \dots \wedge k_m$ , denote by  $Q_i^\phi$  the non-commutative formula translation  $\text{tr}'(k_i)$  of  $k_i$ , where  $i = 1, 2, \dots, m$ .

► **Theorem 1.7.** For a CNF  $\phi = k_1 \wedge \dots \wedge k_m$  where  $Q_1^\phi, \dots, Q_m^\phi$  are the corresponding non-commutative formulas for the clauses, if there is a non-commutative IPS refutation of size  $s$  of  $Q_1^\phi, \dots, Q_m^\phi$  over  $GF(2)$ , then there is a Frege proof of size  $s^{O(\log s)}$  of  $\neg\phi$ .

The proof of Theorem 1.7 consists of several separate steps of independent interest. Essentially, the argument is a short Frege proof for a *reflection principle* for the non-commutative IPS system (a reflection principle for a given proof system  $P$  is a statement that says that if a formula is provable in  $P$  than the formula is also true). The argument becomes rather complicated because we need to prove properties of the *evaluation procedure* of non-commutative formulas, within the restricted framework of propositional Frege proofs.

The *quasi-polynomial blowup* in Theorem 1.7 depends *solely* on the fact that the reflection principle for non-commutative IPS is efficiently provable (apparently) only when the non-commutative IPS certificate is written as a sum of *homogenous* non-commutative formulas, as we now explain. Note that it is not known whether any arithmetic formula can be turned into a (sum of) homogenous formulas with only a polynomial increase in size (in contrast to the standard efficient homogenization of arithmetic *circuits* by Strassen [33]). Recently Raz [25] showed how to transform an arithmetic formula into (a sum of) homogenous formulas with only a quasi-polynomial increase in size. In Lemma 5.6 we show that:

1. The same construction in [25] holds also for *non-commutative* formulas.
2. This construction for non-commutative formulas can be carried out efficiently inside Frege. That is, if  $F$  is a non-commutative formula of size  $s$  computing a homogenous non-commutative polynomial over  $GF(2)$  and  $F'$  is a homogenous non-commutative formula computing the same polynomial with size  $s^{O(\log s)}$  (existing by [25]), then Frege admits an  $s^{O(\log s)}$  size proof of  $F \equiv F'$ .

Before we homogenize the non-commutative formulas (according to Raz' construction [25]) we need to *balance* them, so that their depth is logarithmic in their size. We inspect that the recent construction of Hrubeš and Wigderson [11], for balancing non-commutative formulas with division gates (incurring at most a polynomial increase in size) results in a *division-free* formula, when the *initial* non-commutative formula is division-free itself. Therefore, we can assume that the non-commutative IPS certificate is already balanced.

To prove Theorem 1.7 we thus start with a non-commutative IPS certificate  $\pi$  over  $GF(2)$  of the polynomial translation of the CNF  $\phi$ , written as a *balanced* non-commutative formula

(over  $GF(2)$ ). We then consider this non-commutative polynomial identity *over  $GF(2)$  as a Boolean tautology* by replacing plus gates with XOR and product gates with AND. We convert this Boolean tautology to a homogenous representation (as described above, using a simulation of [25]). Now, we have a Boolean tautology which we denote by  $\pi$ .

We wish to prove  $\neg\phi$  in Frege, using the fact that  $\pi$  is a (massaged version of a) non-commutative IPS certificate. To this end we essentially construct an efficient Frege proof of the correctness of the Raz and Shpilka non-commutative formulas PIT algorithm [26]. The PIT algorithm in [26] uses some basic linear algebraic concepts that might complicate the proof in Frege. However, since we only need to show the *existence* of short Frege proofs for the PIT algorithm's correctness, we can supply *witnesses* to witness the desired linear algebraic objects needed in the proof (these witnesses will be a sequence of linear transformations).

Furthermore, to reason inside Frege directly about the algorithm of [26] is apparently impossible, since this algorithm first converts a non-commutative formula into an *algebraic branching program* (ABP); but apparently the evaluation of ABPs cannot be done with Boolean formulas (and accordingly Frege possibly cannot reason about the evaluation of ABPs). The reason for this apparent inability of Frege to reason about ABP's evaluation is that an ABP is a "sequential" object (an evaluation of an ABP seems to follow from the source to sink, level by level), while Frege operates with formulas, which are "parallel" objects (evaluation of [balanced] formulas can be done in logarithmic time, in case we have enough [separate] processors to perform parallel sub-evaluations of the formula). To overcome this obstacle we show how to perform Raz and Shpilka's PIT algorithm *directly on non-commutative formulas*, without converting the formulas first into ABPs. This technical contribution takes a large part of the argument. We are thus able to prove the following statement, which might be interesting by itself:

► **Lemma 1.8.** *If a non-commutative homogeneous formula  $F(\bar{x})$  over  $GF(2)$  of size  $s$  is identically zero, then the corresponding Boolean formula  $\neg F_{\text{bool}}(\bar{x})$  (where  $F_{\text{bool}}$  results by replacing  $+$  with XOR and  $\cdot$  with AND in  $F(\bar{x})$ ) can be proved with a Frege proof of size at most  $s^{O(1)}$ .*

## 1.4 Significance and discussion

The propositional-calculus is one of the most natural and central notions in logic, and within proof complexity it has a dominant role as a strong proof system whose structure and complexity is poorly understood. In that respect, our characterization of Frege proofs (and thus propositional-calculus) simply as non-commutative polynomials whose non-commutative formula size corresponds (up to a quasi-polynomial factor) to the size of Frege proofs, should be considered a valuable contribution. Since non-commutative formulas constitute a weak model of computation that is quite well understood, and since the Frege system is considered a strong proof system, and it is not entirely out of the way that Frege – or at least its extension, Extended Frege – is polynomially bounded (i.e., admits polynomial-size proofs for every tautology), our results showing the correspondence between Frege proofs and non-commutative formulas are quite surprising.

This correspondence, between non-commutative formulas and proofs, also gives renewed hope for progress on the major fundamental lower bounds problems in proof complexity: it reduces the problem of proving lower bounds on Frege proofs to the problem of establishing rank lower bounds on matrices associated with non-commutative polynomials, where the non-commutative polynomials are given "semi-explicitly" (that is, they are given in terms of the properties of the non-commutative IPS (Definition 1.2)). It is already known that rank lower

bounds yielding strong non-commutative formulas lower bounds are fairly simple (cf. [20]). This then provides a quite compelling evidence that Frege lower bounds, although mostly considered out of reach of current techniques, might not be very far away. Furthermore, our result simplifies greatly the high level-lower bound approach laid out in [10]: the suggested lower bound approach in [10] proposed to move from (commutative) arithmetic circuits lower bounds towards proof complexity lower bounds; but for (commutative) arithmetic circuits there are no known explicit lower bounds, in contrast to non-commutative formulas which constitute a well understood circuit class: both explicit exponential lower bounds and a deterministic PIT algorithms are known for non-commutative formulas.

The new characterization of Frege proofs also sheds light on the correspondence between *circuits and proofs* in proof complexity: in the framework of the ideal proof system, a Frege proof can be seen from the computational perspective as a non-commutative formula.

We also tighten the results of [10]. Namely, by showing that already the non-commutative version of the IPS is sufficient to simulate Frege. As well as by showing *unconditional* efficient simulation of the non-commutative IPS by Frege.

While proving that Frege quasi-polynomially simulates the non-commutative IPS, we demonstrate new simulations of algebraic complexity constructions within proof complexity; these include the homogenization for formulas of Raz [25] and the PIT algorithm for non-commutative formulas by Raz and Shpilka [26]. These proof complexity simulations adds to the known previous such simulations shown in Hrubeš and Tzameret [13] and might be interesting by themselves.

Lastly, this work emphasizes the importance and usefulness of non-commutative models of computation in proof complexity (see [12, 18] for more on this).

## 1.5 Comparison with previous work

As discussed before, our main characterization of the Frege system is based on a non-commutative version of the IPS system from Grochow and Pitassi [10]. As described above, the non-commutative IPS gives a tighter characterization than the (commutative) IPS in [10]. Thus, our proof system is seemingly weaker than the original (formula version of) IPS, and hence apparently *closer to capture the Frege system*.

Proofs in the original (formula version of the) IPS are arithmetic formulas, and thus any super-polynomial lower bound on IPS refutations implies  $VNP \neq VP_e$ , or in other words, that the permanent does not have polynomial-size arithmetic formulas (Joshua Grochow [personal communication]). This gives a justification of the considerable hardness of proving IPS lower bounds. On the other hand, an exponential-size lower bound on our non-commutative IPS gives only a corresponding lower bound on non-commutative formulas, for which exponential-size lower bounds are already known [20]. Since Frege is quasi-polynomially equivalent to the non-commutative IPS, this means that exponential-size lower bounds on Frege implies merely – at least in the context of the Ideal Proof System – corresponding lower bounds on non-commutative formulas, a result which is however already known. This implies again that there is no strong concrete justification to believe that Frege lower bounds are beyond current techniques.

The work in [35] dealt with propositional proof systems over non-commutative formulas. The difference with the current work is that [35] formulated all proof systems as variants of the polynomial calculus and hence the characterization of a proof system in terms of a *single* non-commutative polynomial is lacking from that work (as well as the consequences we obtained in the current work).



## 2 Preliminaries

### 2.1 Frege proof systems

► **Definition 2.1** (Boolean formula). Given a set of input variables  $x_1, \dots, x_n$ , a *Boolean formula* on the inputs is a rooted tree of fan-in at most two, with edges directed from leaves to the root. Internal nodes are labeled with the Boolean gates  $\vee, \wedge, \neg$ , and the fan-in of  $\vee, \wedge$  is two and the fan-in of  $\neg$  is one. The leaves are labeled either with input variables or with 0, 1 (identified with the truth values *false* and *true*, resp.). The entire formula computes the function computed by the gate at the root. Given a formula  $F$ , the *size* of the formula is the number of Boolean gates in  $F$ .

Informally, a Frege proof system is just a standard propositional proof system for proving propositional tautologies (one learns in a basic logic course), having axioms and deduction rules, where proof-lines are written as Boolean formulas. The *size* of a Frege proof is the number of symbols it takes to write down the proof.

The problem of demonstrating super-polynomial size lower bounds on propositional proofs (called also Frege proofs) asks whether there is a family  $(F_n)_{n=1}^\infty$  of propositional tautological formulas for which there is no polynomial  $p$  such that the minimal Frege proof size of  $F_n$  is at most  $p(|F_n|)$ , for all  $n \in \mathbb{Z}^+$  (where  $|F_n|$  denotes the size of the formula  $F_n$ ).

► **Definition 2.2** (Frege rule). A *Frege rule* is a sequence of propositional formulas  $A_0(\bar{x}), \dots, A_k(\bar{x})$ , for  $k \geq 0$ , written as  $\frac{A_1(\bar{x}), \dots, A_k(\bar{x})}{A_0(\bar{x})}$ . In case  $k = 0$ , the Frege rule is called an *axiom scheme*. A formula  $F_0$  is said to be *derived by the rule* from  $F_1, \dots, F_k$  if  $F_0, \dots, F_k$  are all substitution instances of  $A_1, \dots, A_k$ , for some assignment to the  $\bar{x}$  variables (that is, there are formulas  $B_1, \dots, B_n$  such that  $F_i = A_i(B_1/x_1, \dots, B_n/x_n)$ , for all  $i = 0, \dots, k$ ). The Frege rule is said to be *sound* if whenever an assignment satisfies the formulas in the upper side  $A_1, \dots, A_k$ , then it also satisfies the formula in the lower side  $A_0$ .

► **Definition 2.3** (Frege proof). Given a set of Frege rules, a *Frege proof* is a sequence of Boolean formulas such that every proof-line is either an axiom or was derived by one of the given Frege rules from previous proof-lines. If the sequence terminates with the Boolean formula  $A$ , then the proof is said to be a *proof* of  $A$ . The *size* of a Frege proof is the total sizes of all the Boolean formulas in the proof.

A proof system is said to be *implicationaly complete* if for all set of formulas  $T$ , if  $T$  semantically implies  $F$ , then there is a proof of  $F$  using (possibly) axioms from  $T$ . A proof system is said to be *sound* if it admits proofs of only tautologies (when not using auxiliary axioms, like in the  $T$  above).

► **Definition 2.4** (Frege proof system). Given a propositional language and a set  $P$  of sound Frege rules, we say that  $P$  is a *Frege proof system* if  $P$  is implicationaly complete.

Note that a Frege proof is always sound since the Frege rules are assumed to be sound. We do not need to work with a specific Frege proof system, since a basic result in proof complexity states that every two Frege proof systems, even over different languages, are polynomially equivalent [29].

### 2.2 Algebraic proof systems

In this section, we give the definitions the algebraic proof systems Polynomial Calculus over Formulas ( $\mathcal{F}\text{-}\mathcal{PC}$ ) defined by Grigoriev and Hirsch [9]. We start with the definition of a non-commutative formula:



► **Definition 2.5** (Non-commutative formula). Let  $\mathbb{F}$  be a field and  $x_1, x_2, \dots$  be variables. A *noncommutative arithmetic formula* (or *noncommutative formula* for short) is a labeled tree, with edges directed from the leaves to the root, and with fan-in at most two, such that there is an order on the edges coming into a node (the first edge is called the *left* edge and the second one the *right* edge). Every leaf of the tree (namely, a node of fan-in zero) is labeled either with an input variable  $x_i$  or a field  $\mathbb{F}$  element. Every other node of the tree is labeled either with  $+$  or  $\times$  (in the first case the node is a plus gate and in the second case a product gate). We assume that there is only one node of out-degree zero, called *the root*. A noncommutative formula *computes* a noncommutative polynomial in  $\mathbb{F}\langle x_1, \dots, x_n \rangle$  in the following way. A leaf computes the input variable or field element that labels it. A plus gate computes the sum of polynomials computed by its incoming nodes. A product gate computes the *noncommutative* product of the polynomials computed by its incoming nodes according to the order of the edges. (Subtraction is obtained using the constant  $-1$ .) The output of the formula is the polynomial computed by the root. The *depth* of a formula is the maximal length of a path from the root to the leaf. The *size* of a noncommutative formula  $f$  is the total number of nodes in its underlying tree, and is denoted  $|f|$ .

The definition of (a commutative) arithmetic formula is almost identical:

► **Definition 2.6** (Arithmetic formula). An *arithmetic formula* is defined in a similar way to a noncommutative formula, except that we ignore the order of multiplication (that is, a product node does not have order on its children and there is no order on multiplication when defining the polynomial computed by a formula).

Given a pair of non-commutative formulas  $F$  and  $G$  and a variable  $x_i$ , we denote by  $F[G/x_i]$  the formula  $F$  in which every occurrence of  $x_i$  is substituted by the formula  $G$ .

Note that an arithmetic formula is a syntactic object. For example,  $x_1 + x_2$  and  $x_2 + x_1$  are different formulas because commutativity might not hold (even if commutativity holds, we will regard them as different formulas. And in the proof system  $\mathcal{F}\text{-}\mathcal{PC}$  they can be *derived* from each other via the “commutativity of addition”).

### 2.2.1 Polynomial calculus over formulas F-PC system

The  $\mathcal{F}\text{-}\mathcal{PC}$  proof system defined by Grigoriev and Hirsch [9] operates with arithmetic formulas (as purely syntactic terms).

► **Definition 2.7** ( $\mathcal{F}\text{-}\mathcal{PC}$  [9]). Fix a field  $\mathbb{F}$ . Let  $F := \{f_1, \dots, f_m\}$  be a collection of *formulas* computing polynomials from  $\mathbb{F}[x_1, \dots, x_n]$  (note here that we are talking about formulas (treated as syntactic terms), and *not* polynomials. Also notice that all formulas in  $\mathcal{F}\text{-}\mathcal{PC}$  are (commutative) formulas computing (commutative) polynomials). Let the set of axioms be the following formulas:

**Boolean axioms**  $x_i \cdot (1 - x_i)$ , for all  $1 \leq i \leq n$ .

A sequence  $\pi = (\Phi_1, \dots, \Phi_\ell)$  of formulas computing polynomials from  $\mathbb{F}[x_1, \dots, x_n]$  is said to be an  $\mathcal{F}\text{-}\mathcal{PC}$  proof of  $\Phi_\ell$  from  $F$ , if for every  $i \in [\ell]$  we have one of the following:

1.  $\Phi_i = f_j$ , for some  $j \in [m]$ ;
2.  $\Phi_i$  is a Boolean axiom;
3.  $\Phi_i$  was deduced by one of the following inference rules from previous proof-lines  $\Phi_j, \Phi_k$ , for  $j, k < i$ :

**Product**

$$\frac{\Phi}{x_r \cdot \Phi}, \quad \text{for } r \in [n].$$

**Addition**

$$\frac{\Phi \quad \Theta}{a \cdot \Phi + b \cdot \Theta}, \quad \text{for } a, b \in \mathbb{F}.$$

(Where  $\Phi, x_r \cdot \Phi, \Theta, a \cdot \Phi, b \cdot \Theta$  are *formulas* constructed as displayed; e.g.,  $x_r \cdot \Phi$  is the formula with product gate at the root having the formulas  $x_r$  and  $\Phi$  as children.) (In [9] the product rule of  $\mathcal{F}\text{-PC}$  is defined so that one can derive  $\Theta \cdot \Phi$  from  $\Phi$ , where  $\Theta$  is any formula, and not just a variable. However, the definition of  $\mathcal{F}\text{-PC}$  in [9] and our Definition 2.7 polynomially-simulate each other.)

4.  $\Phi_i$  was deduced from previous proof-line  $\Phi_j$ , for  $j < i$ , by one of the following *rewriting rules* expressing the polynomial-ring axioms (where  $f, g, h$  range over all arithmetic formulas computing polynomials in  $\mathbb{F}[x_1, \dots, x_n]$ ):

**Zero rule**  $0 \cdot f \leftrightarrow 0$

**Unit rule**  $1 \cdot f \leftrightarrow f$

**Scalar rule**  $t \leftrightarrow \alpha$ , where  $t$  is a formula containing no variables (only field  $\mathbb{F}$  elements) that computes the constant  $\alpha \in \mathbb{F}$ .

**Commutativity rules**  $f + g \leftrightarrow g + f, \quad f \cdot g \leftrightarrow g \cdot f$

**Associativity rule**  $f + (g + h) \leftrightarrow (f + g) + h, \quad f \cdot (g \cdot h) \leftrightarrow (f \cdot g) \cdot h$

**Distributivity rule**  $f \cdot (g + h) \leftrightarrow (f \cdot g) + (f \cdot h)$

(The semantics of an  $\mathcal{F}\text{-PC}$  proof-line  $p_i$  is the polynomial equation  $p_i = 0$ .) An  $\mathcal{F}\text{-PC}$  *refutation* of  $F$  is a proof of the formula  $1$  from  $F$ . The *size* of an  $\mathcal{F}\text{-PC}$  proof  $\pi$  is defined as the total size of all formulas in  $\pi$  and is denoted by  $|\pi|$ .

► **Definition 2.8** (Tree-like  $\mathcal{F}\text{-PC}$ ). A system  $\mathcal{F}\text{-PC}$  is a *tree-like  $\mathcal{F}\text{-PC}$*  if every derived arithmetic formula in the proof system is used only once (and if it is needed again, it must be derived once more).

### 2.2.1.1 Translation of Boolean formulas into polynomial equations

The proof system  $\mathcal{F}\text{-PC}$  can be considered as a propositional proof system for Boolean tautologies (namely, Boolean formulas that are true under any assignment). Given a Boolean formula  $T$  in the propositional variables  $x_1, \dots, x_n$  we can transform  $T$  into a set of polynomial equations by encoding it into a set of arithmetic formulas where each clause in the CNF corresponds to an arithmetic formula by replacing  $\wedge$  with  $\times$ ,  $\vee$  with  $+$  and  $\neg x$  with  $1 - x$ ; and for each variable  $x_i$ , add  $x_i^2 - x_i$  (called the *Boolean axioms*) to guarantee that every satisfying assignment to the variables is a 0-1 assignment. Then the given CNF is a tautology if and only if the set of arithmetic formulas have no common root.

► **Definition 2.9** (Polynomially Simulation). Let  $\mathcal{P}_1, \mathcal{P}_2$  be two proof systems for the same language  $L$  (in case the proof systems are for two different languages we fix a translation from one language to the other, as described above). We say that  $\mathcal{P}_2$  *polynomially simulates*  $\mathcal{P}_1$  if given a  $\mathcal{P}_1$  proof (or refutation)  $\pi$  of a  $F$ , then there exists a proof (respectively, refutation) of  $F$  in  $\mathcal{P}_2$  of size polynomial in the size of  $\pi$ . In case  $\mathcal{P}_2$  polynomially simulates  $\mathcal{P}_1$  while  $\mathcal{P}_1$  does not polynomially simulates  $\mathcal{P}_2$  we say that  $\mathcal{P}_2$  is *strictly stronger* than  $\mathcal{P}_1$ .

In [9], it was shown that  $\mathcal{F}\text{-PC}$ , as well as tree-like  $\mathcal{F}\text{-PC}$ , polynomially simulate Frege. We repeat the argument for the convenience of the reader:

► **Theorem 2.10** ([9]). *Tree-like  $\mathcal{F}\text{-PC}$  polynomially simulates Frege.*

**Proof.** The following was shown in [9]:

► **Theorem 2.11** (Theorem 3, [9]). *The system  $\mathcal{F}\text{-PC}$  polynomially simulates the Frege system.*

Moreover, inspecting the proof of the above theorem, we can observe that tree-like Frege proofs are simulated by tree-like  $\mathcal{F}\text{-PC}$  proofs:

► **Lemma 2.12.** *Tree-like  $\mathcal{F}\text{-PC}$  polynomially simulates tree-like Frege systems.*

But Krajíček showed that tree-like Frege and Frege are polynomially equivalent:

► **Theorem 2.13** ([14]). *Tree-like Frege proofs polynomially simulate Frege proofs.*

Thus, by this theorem and by Lemma 2.11, tree-like  $\mathcal{F}\text{-PC}$  polynomially simulates the Frege system. ◀

### 3 The non-commutative ideal proof system

The non-commutative ideal proof system (non-commutative IPS for short) is an algebraic refutation system in which a refutation is a single non-commutative polynomial. In the next section we show that when the non-commutative IPS refutations are written as *non-commutative formulas* then the non-commutative IPS polynomially simulates tree-like  $\mathcal{F}\text{-PC}$ , and hence polynomially simulates the Frege proof system (by [9]).

► **Definition 3.1** (Non-commutative IPS). Let  $\mathbb{F}$  be a field. Assume that  $F_1(\bar{x}) = F_2(\bar{x}) = \dots = F_m(\bar{x}) = 0$  is a system of non-commutative polynomial equations from  $\mathbb{F}\langle x_1, \dots, x_n \rangle$ , and suppose that the following set of equations (axioms) are included in the  $F_i(\bar{x})$ 's:

**Boolean axiom:**  $x_i \cdot (1 - x_i)$ , for all  $1 \leq i \leq n$ ;

**Commutator axiom:**  $x_i \cdot x_j - x_j \cdot x_i$ , for all  $1 \leq i < j \leq n$ .

Suppose that the  $F_i(\bar{x})$ 's have no common 0-1 solutions. (One can check that the  $F_i(\bar{x})$ 's have no common 0-1 solutions in  $\mathbb{F}$  iff they do not have a common 0-1 solution in every  $\mathbb{F}$ -algebra.) A *non-commutative IPS refutation* (or *certificate*) that the system of  $F_i(\bar{x})$ 's is unsatisfiable is a **non-commutative polynomial**  $\mathfrak{F}(\bar{x}, \bar{y})$  in the variables  $x_1, \dots, x_n$  and  $y_1, \dots, y_m$  (i.e.  $\mathfrak{F} \in \mathbb{F}\langle \bar{x}, \bar{y} \rangle$ ), such that:

1.  $\mathfrak{F}(x_1, \dots, x_n, \bar{0}) = 0$ ; and
2.  $\mathfrak{F}(x_1, \dots, x_n, F_1(\bar{x}), \dots, F_m(\bar{x})) = 1$ .

In this paper we assume that the non-commutative IPS refutation is written as a *non-commutative formula*. Hence the **size** of a non-commutative IPS refutation is the minimal size of a non-commutative formula computing the non-commutative IPS refutation.

- **Comment. 1.** The identities in items 1 and 2 in Definition 3.1 are *formal* identities of polynomials (i.e., in 1 the polynomial in the left hand side has a zero coefficient for every monomial, and in 2 the only nonzero monomial is the monomial 1).
2. In order to prove that a system of *commutative* polynomial equations  $\{P_i = 0\}$  (where each  $P_i$  is expressed as an arithmetic formula) has no common roots in non-commutative IPS, we write each  $P_i$  as a *non-commutative formula* (in some way; note that there is no unique way to do this).
  3. When we write  $P \cdot Q - Q \cdot P$  where  $P, Q$  are formulas (e.g.,  $x_i$  and  $x_j$ , resp.), we mean  $((P \cdot Q) + (-1 \cdot (Q \cdot P)))$ .

## 4 Non-commutative ideal proof system polynomially simulates Frege

Here we show that the non-commutative IPS polynomially simulates Frege.

► **Theorem 4.1** (restatement of Theorem 1.4). *The non-commutative IPS refutation system, when refutations are written as non-commutative formulas, polynomially simulates Frege systems. More precisely, for every propositional tautology  $T$ , if  $T$  has a polynomial-size Frege proof then there is a non-commutative IPS certificate (with integer coefficients) of polynomial non-commutative formula size.*

Recall that Raz and Shpilka [26] gave a deterministic polynomial-time PIT algorithm for non-commutative formulas (over any field):

► **Theorem 4.2** (PIT for non-commutative formulas [26]). *There is a deterministic polynomial-time algorithm that decides whether a given noncommutative formula over a field  $\mathbb{F}$  computes the zero polynomial 0. (We assume here that the field  $\mathbb{F}$  can be efficiently represented (e.g., the field of rationals).)*

Now, since we write refutations as non-commutative formulas we can use the theorem above to check in *deterministic* polynomial-time the correctness of non-commutative IPS refutations, obtaining:

► **Corollary 4.3** (restatement of Corollary 1.5). *The non-commutative IPS is a sound and complete Cook-Reckhow refutation system. That is, it is a sound and complete refutation system for unsatisfiable propositional formulas in which refutations can be checked for correctness in deterministic polynomial-time.*

To prove Theorem 4.1, we will show in Section 4.1 that the non-commutative IPS polynomially-simulates tree-like  $\mathcal{F}\text{-PC}$  (Definition 2.7), which suffices to complete the proof due to Theorem 2.10.

### 4.1 Non-commutative IPS polynomially simulates tree-like F-PC

For convenience, let  $C_{i,j}$  denote the commutator axiom  $x_i \cdot x_j - x_j \cdot x_i$ , for  $i, j \in [n], i \neq j$ .

► **Theorem 4.4.** *Non-commutative IPS polynomially simulates Tree-like  $\mathcal{F}\text{-PC}$  (Definition 2.7).*

**Proof.** Let  $F_1, \dots, F_m$  be arithmetic formulas over the variables  $x_1, \dots, x_n$ . Note that an arithmetic formula is a syntactic term in which the children of gates are ordered. We thus can treat a (commutative) arithmetic formula as a *non-commutative* arithmetic formula by taking the *order* on the children of products gates to be the order of non-commutative multiplication.

Suppose  $\mathcal{F}\text{-PC}$  has a  $\text{poly}(n)$ -size tree-like refutation  $\pi := (L_1, \dots, L_k)$  of the  $F_i$ 's (i.e., a proof of the polynomial 1 from  $F_1, \dots, F_m$ ), where each  $L_j$  is an arithmetic formula. We construct a corresponding non-commutative IPS refutation of the  $F_i$ 's from this  $\mathcal{F}\text{-PC}$  tree-like refutation. Denote by  $|\pi|$  the size of  $\pi$ . We have the following:

► **Lemma 4.5.** *For each  $i \in [k]$ , there exists a non-commutative formula  $\phi_i$  such that:*

1.  $\phi_i(\bar{x}, \bar{0}) = 0$ ;
2.  $\phi_i(\bar{x}, F_t, C_{j,j'}) = L_i$ , where  $t \in [m]$ ,  $j, j' \in [n]$ ,  $j < j'$ ; (this is an abuse of notation meaning  $\phi_i(\bar{x}, F_1, \dots, F_m, C_{1,2}, C_{1,3}, \dots, C_{n-1,n})$ ). We use a similar abuse of notation in the sequel.)

3.  $|\phi_i| \leq \left(\sum_{\ell \in A_i} |L_\ell|\right)^4$ , where  $A_i \subset [k]$  refers to the indices of the  $\mathcal{F}$ - $\mathcal{PC}$  proof-lines involved in deriving  $L_i$ . (For example, if  $L_i$  is derived by  $L_\alpha$  and  $L_\alpha$  is derived by  $L_\beta$  for some  $\beta < \alpha < i \in [k]$ , then we say that  $\alpha, \beta$  are both involved for deriving  $L_i$ .)

Note that if the lemma holds, then  $\phi_k$  will be a non-commutative IPS proof because it has the property that  $\phi_k(\bar{x}, \bar{0}) = 0$  and  $\phi_k(\bar{x}, F_t, C_{j,j'}) = L_k = 1$ , where  $t \in [m], j, j' \in [n], j \neq j'$ . And its size is bounded by  $\left(\sum_{\ell \in A_k} |L_\ell|\right)^4 \leq \left(\sum_{\ell \in [k]} |L_\ell|\right)^4 \leq O(|\pi|^4)$ . Thus, non-commutative IPS polynomially simulates tree-like  $\mathcal{F}$ - $\mathcal{PC}$ .  $\blacktriangleleft$

We construct  $\phi_i$  by induction on the length  $k$  of the refutation  $\pi$ . That is, for  $i$  from 1 to  $k$ , we construct the non-commutative formula  $\phi_i(\bar{x}, \bar{y})$  according to  $L_i$ , as follows:

#### Case 1:

The  $L_i$  is the input axiom  $F_j$  for some  $j \in [m]$ .

Let  $\phi_i := y_j$ . Obviously,  $\phi_i(\bar{x}, 0) = 0, \phi_i(\bar{x}, F_t, C_{\alpha,\beta}) = F_j = L_i$  and  $|\phi_i| = 1 \leq |L_i|^4$ .

#### Case 2:

The  $L_i$  is derived from an inference rule from previous proof-lines  $L_j, L_{j'}$ , for  $j, j' < i$ . Then we divide this case into two parts.

Part (1): The  $L_i$  is derived from the addition rule  $L_i = aL_j + bL_{j'}$ . Put  $\phi_i := a\phi_j + b\phi_{j'}$  where  $a, b \in \mathbb{F}$ . Thus,  $\phi_i(\bar{x}, 0) = a\phi_j(\bar{x}, 0) + b\phi_{j'}(\bar{x}, 0) = 0, \phi_i(\bar{x}, F_t, C_{\alpha,\beta}) = aL_j + bL_{j'} = L_i$  and  $|\phi_i| = |\phi_j| + |\phi_{j'}| + 3 \leq \left(\sum_{\ell \in A_j} |L_\ell|\right)^4 + \left(\sum_{\ell \in A_{j'}} |L_\ell|\right)^4 + 3 \leq \left(\sum_{\ell \in A_i} |L_\ell|\right)^4$  (where the right most inequality holds since  $\pi$  is a *tree-like* refutation and hence  $A_j \cap A_{j'} = \emptyset$ ).

Part (2): The  $L_i$  is derived from the product rule  $L_i = x_r \cdot L_{j'}$  for  $r \in [n]$ . Put  $\phi_i := (x_r \cdot \phi_{j'})$ . Then  $\phi_i(\bar{x}, 0) = x_r \cdot \phi_{j'}(\bar{x}, 0) = 0, \phi_i(\bar{x}, F_t, C_{\alpha,\beta}) = x_r \cdot L_{j'} = L_i$  and  $|\phi_i| = |\phi_{j'}| + 2 \leq \left(\sum_{\ell \in A_{j'}} |L_\ell|\right)^4 + 2 \leq \left(\sum_{\ell \in A_i} |L_\ell|\right)^4$ .

#### Case 3:

The  $L_i$  is derived from  $L_j$  by a rewriting rule excluding the commutative rule of multiplication. Let  $\phi_i := \phi_j$ . The non-commutative  $\phi_i$  satisfies the properties claimed trivially since all the rewriting rules (excluding the commutative rule of multiplication) express the non-commutative polynomial-ring axioms, and thus cannot change the polynomial computed by a non-commutative formula. And  $|\phi_i| = |\phi_j| \leq \left(\sum_{\ell \in A_i} |L_\ell|\right)^4$ .

#### Case 4:

The  $L_i$  is derived from  $L_j$  by a single application of the commutative rule of multiplication. Then by Lemma 4.6 below, we can construct a non-commutative formula  $\phi_{L_i, L_j}$  such that  $\phi_i := (\phi_j + \phi_{L_i, L_j})$  satisfies the desired properties (stated in Lemma 4.5).

► **Lemma 4.6.** *Let  $L_i, L_j$  be non-commutative formulas such that  $L_i$  can be derived from  $L_j$  via the commutative rule of multiplication. Then there is a non-commutative formula  $\phi_{L_i, L_j}(\bar{x}, \bar{y})$  in variables  $\{x_\ell, y_{\alpha,\beta}, \ell \in [n], \alpha < \beta \in [n]\}$ , such that:*

1.  $\phi_{L_i, L_j}(\bar{x}, \bar{0}) = 0$ ;
2.  $\phi_{L_i, L_j}(\bar{x}, C_{\alpha,\beta}) = L_i - L_j$ ;
3.  $|\phi_{L_i, L_j}| \leq |L_i|^2 |L_j|^2$ .

**Proof.** We define the non-commutative formula  $\phi_{L_i, L_j}$  inductively as follows:

- If  $L_i = (P \cdot Q)$ , and  $L_j = (Q \cdot P)$ , then  $\phi_{L_i, L_j}$  is defined to be the formula constructed in Lemma 4.7 below.
- If  $L_i = (P \cdot Q)$ ,  $L_j = (P' \cdot Q')$ .
  - Case 1. If  $P = P'$ , then let  $\phi_{L_i, L_j} := (P \cdot \phi_{Q, Q'})$ .
  - Case 2. If  $Q = Q'$ , then let  $\phi_{L_i, L_j} := (\phi_{P, P'} \cdot Q)$ .
- If  $L_i = (P + Q)$ ,  $L_j = (P' + Q')$ .
  - Case 1. If  $P = P'$ , then let  $\phi_{L_i, L_j} = \phi_{Q, Q'}$ .
  - Case 2. If  $Q = Q'$ , then let  $\phi_{L_i, L_j} = \phi_{P, P'}$ .

By induction, one could check the construction satisfies the desired properties.  $\blacktriangleleft$

► **Lemma 4.7.** *For any pair  $P, Q$  of two non-commutative formulas there exists a non-commutative formula  $F$  in variables  $\{x_\ell, y_{i,j}, \ell \in [n], i < j \in [n]\}$  such that:*

1.  $F(\bar{x}, \bar{0}) = 0$ ;
2.  $F(\bar{x}, C_{i,j}) = P \cdot Q - Q \cdot P$ ;
3.  $|F| = |P|^2 |Q|^2$ .  $e$

**Proof.** Let  $s(P, Q)$  denote the smallest size of  $F$  satisfying the above properties. We will show that  $s(P, Q) \leq |P|^2 \cdot |Q|^2$  by induction on  $\max(|P|, |Q|)$ .

*Base case:*  $|P| = |Q| = 1$ .

In this case both  $P$  and  $Q$  are constants or variables, thus  $s(P, Q) = 1 \leq |P|^2 |Q|^2$ .

In the following induction step, we consider the case that  $|P| \geq |Q|$  (which is symmetric for the case  $|P| < |Q|$ ).

*Induction step:* Assume that  $|P| \geq |Q|$  (the case  $|P| < |Q|$  is similar).

Case 1: The root of  $P$  is addition.

Let  $P = (P_1 + P_2)$ . We have (after rearranging):

$$P \cdot Q - Q \cdot P = ((P_1 \cdot Q - Q \cdot P_1) + (P_2 \cdot Q - Q \cdot P_2))$$

By induction hypothesis, we have  $s(P, Q) \leq s(P_1, Q) + 1 + s(P_2, Q) \leq |P_1|^2 |Q|^2 + 1 + |P_2|^2 |Q|^2 \leq (|P_1| + |P_2| + 1)^2 |Q|^2 = |P|^2 \cdot |Q|^2$ .

Case 2: The root of  $P$  is a product gate.

Let  $P = (P_1 \cdot P_2)$ . By rearranging:

$$P \cdot Q - Q \cdot P = ((P_1 \cdot (P_2 \cdot Q - Q \cdot P_2)) + ((P_1 \cdot Q - Q \cdot P_1) \cdot P_2))$$

By induction hypothesis, we have  $s(P, Q) = |P_1| + 1 + s(P_2, Q) + 1 + s(P_1, Q) + 1 + |P_2| \leq |P_1| + 1 + |P_2|^2 |Q|^2 + 1 + |P_1|^2 |Q|^2 + 1 + |P_2| \leq (|P_1| + |P_2| + 1)^2 |Q|^2 = |P|^2 \cdot |Q|^2$ .  $\blacktriangleleft$

## 5 Frege quasi-polynomially simulates non-commutative IPS

In this section we prove that the Frege system quasi-polynomially simulates the non-commutative IPS (over  $GF(2)$ ). Together with Theorem 4.4, this gives a new characterization (up to a quasi-polynomial increase in size) of propositional Frege proofs as non-commutative arithmetic formulas.

We use the notation in Section 1.3.3: for a clause  $k_i$  in a CNF  $\phi = k_1 \wedge \dots \wedge k_m$ , we denote by  $Q_i^\phi$  the non-commutative formula translation  $\text{tr}'(k_i)$  of the clause  $k_i$  (Definition 1.6). Thus,  $\neg x$  is translated to  $x$ ,  $x$  is translated to  $1 - x$  and  $f_1 \cdots f_r$  is translated to  $\prod_i \text{tr}'(f_i)$

(considered as a tree of product gates with  $\text{tr}'(f_i)$  as leaves), and where the formulas are over  $GF(2)$  (meaning that  $1 - x$  is in fact  $1 + x$ ).

► **Theorem 5.1** (Main quasi-polynomial simulation). *For a 3CNF  $\phi = k_1 \wedge \dots \wedge k_m$  where  $Q_1^\phi, \dots, Q_m^\phi$  are the corresponding polynomial equations for the clauses, if there is a non-commutative IPS refutation of size  $s$  of  $Q_1^\phi, \dots, Q_m^\phi$  over  $GF(2)$ , then there is a Frege proof of size  $s^{O(\log s)}$  of  $\neg\phi$ .*

The rest of the section is dedicated to proving Theorem 5.1. Due to lack of space we refer the reader to the full version of this work [19] for complete proofs. Here we shall only outline the main parts of the proof (see also Section 1.3.3).

## 5.1 Balancing non-commutative formulas

First we show that a non-commutative formula of size  $s$  can be balanced to an equivalent formula of depth  $O(\log s)$ , and thus we can assume that the non-commutative IPS certificate is already given as a balanced formula (this is needed for what follows). Both the statement of the balancing construction and its proof are similar to Proposition 4.1 in Hrubeš and Wigderson [11] (which in turn is similar to the case of commutative formulas with division gates in Brent [5]). Note that a formula of a logarithmic depth must have a polynomial-size. (Thus, in what follows, without loss of generality we will assume that  $F$  is given already in a balanced form, namely has depth  $O(\log s)$  and polynomial-size which, for simplicity, we denoted as  $s$ .)

► **Lemma 5.2.** *Assume that a non-commutative polynomial  $p$  can be computed by a formula of size  $s$ . Then  $p$  can be computed by a formula of depth  $O(\log s)$  (and hence of polynomial-size).*

## 5.2 The reflection principle

Here we show that the existence of a non-commutative IPS refutation of size  $s$  and depth  $O(\log s)$ , implies the existence of a Frege proof with size  $s^{O(\log s)}$  of  $\neg\phi$ . This is done by proving a *reflection principle* for the non-commutative IPS system in Frege. As mentioned in the introduction, informally, a reflection principle for a given proof system  $P$  is a statement that says that if a formula is provable in  $P$  then the formula is also *true*. Thus, suppose we have a short Frege-proof of the reflection principle for  $P$ , having the form:

$$“([\pi] \text{ is a } P\text{-proof of } [T]) \longrightarrow T”,$$

where  $[T]$  and  $[\pi]$  are some reasonable encodings of the tautology  $T$  and its  $P$ -proof  $\pi$ , respectively. Then, we can easily obtain a Frege proof of  $T$  assuming we have a  $P$ -proof of  $T$ .

Let  $F$  be a non-commutative formula over  $GF(2)$  and let  $\overline{Q}^\phi(\overline{x})$  denote the vector  $(Q_1^\phi, \dots, Q_m^\phi)$  (see Theorem 5.1). First, note that  $F$  is a non-commutative IPS proof of  $\phi$  only if it has the following two properties:

$$F(\overline{x}, \overline{0}) = 0, \quad F(\overline{x}, \overline{Q}^\phi(\overline{x})) = 1, \quad (1)$$

showing the unsatisfiability of  $\overline{Q}^\phi(\overline{x}) = 0$ , and hence showing  $\neg\phi$  is a tautology. We can treat  $F$  as a Boolean formula, as follows:

► **Definition 5.3** (Booleanization  $F_{bool}$ ). Let  $F(\overline{x})$  be a non-commutative formula over  $GF(2)$  in the (algebraic) variables  $\overline{x}$ . We denote by  $F_{bool}(\overline{p})$  the Boolean formula in the (propositional) variables  $\overline{p}$  obtained by turning every plus gate and multiplication gate to  $\oplus$  (i.e., XOR)



and  $\wedge$  gates, respectively, and turning the input variables  $\bar{x}$  into the propositional variables  $\bar{p}$ . We sometimes write  $F$  and  $F_{bool}$  without explicitly mentioning the  $\bar{x}$  and  $\bar{p}$  variables, respectively.

When we consider  $F = F(\bar{x}, \bar{y})$  (with both the  $\bar{x}$  and  $\bar{y}$  variables),  $F_{bool}$  denotes the Booleanization of  $F$  when the variables  $\bar{x}$  are replaced by  $\bar{p}$  and the variables  $\bar{y}$  are still written as  $\bar{y}$ . Note that for any 0-1 assignment,  $F$  and  $F_{bool}$  have the same value. Therefore, by the properties in (1), we know:

$$\neg F_{bool}(\bar{p}, \bar{0}), \quad F_{bool}(\bar{p}, \overline{Q_{bool}^\phi}(\bar{p})) \tag{2}$$

are both *tautologies* (though we still need to prove that their Frege proofs are short).

To conclude Theorem 5.1, we first prove in Frege  $\neg\phi$  based on (2) (this is done in Lemma 5.4 below which is not hard to establish), and then we show that there exists an  $s^{O(\log s)}$  Frege proof of (2) (which is done in Lemma 5.5 in the next section, and requires much more work).

► **Lemma 5.4.**  $\left( (\neg F_{bool}(\bar{p}, \bar{0})) \wedge F_{bool}(\bar{p}, \overline{Q_{bool}^\phi}(\bar{p})) \right) \rightarrow \neg\phi$  can be proved with a polynomial-size Frege proof.

Having this lemma, it remains to show a quasi-polynomial-size proof of (2). We denote  $\neg F_{bool}(\bar{p}, \bar{0})$  and  $\neg \left( 1 \oplus F_{bool}(\bar{p}, \overline{Q_{bool}^\phi}(\bar{p})) \right)$  by

$$F'_{bool}(\bar{p}), \quad F''_{bool}(\bar{p}), \quad \text{respectively.} \tag{3}$$

Note that the substitutions of the constants 0 or the constant depth formulae  $\overline{Q_{bool}^\phi}$  in  $F$  cannot increase the depth of  $F$  too much (i.e., can add at most a constant to the size of  $F$ ). In other words, the depths of the formulae in (3) are still  $O(\log s)$ .

### 5.3 Non-commutative formula identities have quasi-polynomial-size proofs

Recall that a (commutative or non-commutative) multivariate polynomial  $f$  is *homogeneous* if every monomial in  $f$  has the same total degree. For each  $0 \leq j \leq d$ , denote by  $f^{(j)}$  the homogenous part of degree  $j$  of  $f$ , that is, the sum of all monomials (together with their coefficient from the field) in  $f$  of total degree  $j$ . We say that a formula is *homogeneous* if each of its gates computes a *homogeneous* polynomial (see Definition 2.5 for the definition of a polynomial computed by a gate or a formula).

To complete the proof of Theorem 5.1 it remains to prove the following:

► **Lemma 5.5.** *If a non-commutative formula  $F(\bar{x})$  with 0-1 coefficients of size  $s$  and depth  $O(\log s)$  is identically zero, then the corresponding Boolean formula  $\neg F_{bool}(\bar{p})$  admits a Frege proof of size  $s^{O(\log s)}$ .*

### 5.4 Homogenization of non-commutative formulas has short Frege proofs

To complete the proof of Lemma 5.5 it remains to prove Lemmas 5.6 and 1.8 in what follows. Lemma 5.6 states that Raz' construction from [25] for homogenizing arithmetic formulas is efficiently provable in Frege (and is also applicable to non-commutative formulas):

► **Lemma 5.6.** *If  $F$  is a non-commutative formula of size  $s$  and depth  $O(\log s)$  and  $F^{(0)}, \dots, F^{(s)}$  are the homogenous formulae computing  $F$ 's homogenous parts of degrees  $0, \dots, s$ , respectively, constructed according to [25], then there exists an  $s^{O(\log s)}$ -size Frege proof of:*

$$\bigoplus_{i=0}^{s+1} F^{(i)} \leftrightarrow F_{\text{bool}}.$$

## 5.5 Homogenous non-commutative formula identities have polynomial-size Frege proofs

To conclude Theorem 5.1 it remains to prove Lemma 5.7 below, which is the main technical lemma of the whole argument. It states that a non-commutative syntactic-homogenous formula identity over  $GF(2)$  has polynomial-size Frege proofs (considered as a Boolean tautology). The proof of this lemma is somewhat lengthy as it entails us to show that the Raz and Shpilka polynomial-time PIT algorithm for non-commutative formulas can be “simulated” efficiently with Frege proofs. Here we just state formally Lemma 1.8 and refer the reader to the full version of the paper [19] for a complete proof of this lemma.

► **Lemma 5.7** (Main technical lemma). *There exists a constant  $c$  such that if a non-commutative syntactic homogeneous formula  $F(\bar{x})$  over  $GF(2)$  of size  $s$  is identically zero, then the corresponding Boolean tautology  $\neg F_{\text{bool}}(\bar{p})$  can be proved with a Frege proof of size at most  $s^c$  (for sufficiently large  $s$ ).*

**Acknowledgements.** We thank Joshua Grochow for helpful comments.

---

### References

- 1 Miklós Ajtai. The complexity of the pigeonhole principle. In *Proceedings of the IEEE 29th Annual Symposium on Foundations of Computer Science*, pages 346–355, 1988.
- 2 Albert Atserias, Nicola Galesi, and Pavel Pudlák. Monotone simulations of non-monotone proofs. *J. Comput. System Sci.*, 65(4):626–638, 2002. Special issue on complexity, 2001 (Chicago, IL).
- 3 Albert Atserias, Phokion G. Kolaitis, and Moshe Y. Vardi. Constraint propagation as a proof system. In *CP*, pages 77–91, 2004.
- 4 Maria Luisa Bonet, Samuel R. Buss, and Toniann Pitassi. Are there hard examples for Frege systems? In *Feasible mathematics, II (Ithaca, NY, 1992)*, volume 13 of *Progr. Comput. Sci. Appl. Logic*, pages 30–56. Birkhäuser Boston, Boston, MA, 1995.
- 5 Richard P. Brent. The parallel evaluation of general arithmetic expressions. *J. ACM*, 21(2):201–206, 1974.
- 6 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 the Theory of Computing (Philadelphia, PA, 1996)*, pages 174–183, New York, 1996. ACM.
- 7 Stephen Cook and Phuong Nguyen. *Logical Foundations of Proof Complexity*. ASL Perspectives in Logic. Cambridge University Press, 2010.
- 8 Stephen A. Cook and Robert A. Reckhow. The relative efficiency of propositional proof systems. *The Journal of Symbolic Logic*, 44(1):36–50, 1979.
- 9 Dima Grigoriev and Edward A. Hirsch. Algebraic proof systems over formulas. *Theoret. Comput. Sci.*, 303(1):83–102, 2003. Logic and complexity in computer science (Créteil, 2001).

- 10 Joshua A. Grochow and Toniann Pitassi. Circuit complexity, proof complexity, and polynomial identity testing. In *55th Annual IEEE Symposium on Foundations of Computer Science, FOCS*, 2014. Also available as arXiv:1404.3820 [cs.CC].
- 11 Pavel Hrubeš and Avi Wigderson. Non-commutative arithmetic circuits with division. In *Innovations in Theoretical Computer Science, ITCS'14, Princeton, NJ, USA, January 12-14, 2014*, pages 49–66, 2014.
- 12 Pavel Hrubeš. How much commutativity is needed to prove polynomial identities? *Electronic Colloquium on Computational Complexity, ECCC*, (Report no.: TR11-088), June 2011.
- 13 Pavel Hrubeš and Iddo Tzameret. Short proofs for the determinant identities. In *Proceedings of the 44th Annual ACM Symposium on the Theory of Computing (STOC)*, New York, 2012. ACM.
- 14 Jan Krajíček. *Bounded arithmetic, propositional logic, and complexity theory*, volume 60 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, Cambridge, 1995.
- 15 Jan Krajíček. An exponential lower bound for a constraint propagation proof system based on ordered binary decision diagrams. *J. Symbolic Logic*, 73(1):227–237, 2008.
- 16 Jan Krajíček, Pavel Pudlák, and Alan Woods. An exponential lower bound to the size of bounded depth Frege proofs of the pigeonhole principle. *Random Structures Algorithms*, 7(1):15–39, 1995.
- 17 Jan Krajíček. *Forcing with random variables and proof complexity*. London Mathematical Society Lecture Note Series, No.382. Cambridge University Press, 2011.
- 18 Fu Li and Iddo Tzameret. Generating matrix identities and proof complexity. *Electronic Colloquium on Computational Complexity, TR13-185*, 2013. arXiv:1312.6242 [cs.CC] <http://arxiv.org/abs/1312.6242>.
- 19 Fu Li, Iddo Tzameret, and Zhengyu Wang. Non-commutative formulas and Frege lower bounds: a new characterization of propositional proofs. 2014. arXiv:1412.8746 [cs.CC] <http://arxiv.org/abs/1412.8746>.
- 20 N. Nisan. Lower bounds for non-commutative computation. *Proceedings of the 23th Annual ACM Symposium on the Theory of Computing*, pages 410–418, 1991.
- 21 Toniann Pitassi, Paul Beame, and Russell Impagliazzo. Exponential lower bounds for the pigeonhole principle. *Comput. Complexity*, 3(2):97–140, 1993.
- 22 Pavel Pudlák. On the complexity of the propositional calculus. In *Sets and proofs (Leeds, 1997)*, volume 258 of *London Math. Soc. Lecture Note Ser.*, pages 197–218. Cambridge Univ. Press, Cambridge, 1999.
- 23 Ran Raz. Separation of multilinear circuit and formula size. *Theory of Computing, Vol. 2, article 6*, 2006.
- 24 Ran Raz. Multi-linear formulas for permanent and determinant are of super-polynomial size. *J. ACM*, 56(2), 2009.
- 25 Ran Raz. Tensor-rank and lower bounds for arithmetic formulas. *J. ACM*, 60(6):40, 2013.
- 26 Ran Raz and Amir Shpilka. Deterministic polynomial identity testing in non commutative models. *Computational Complexity*, 14(1):1–19, 2005.
- 27 Ran Raz and Iddo Tzameret. Resolution over linear equations and multilinear proofs. *Ann. Pure Appl. Logic*, 155(3):194–224, 2008.
- 28 Ran Raz and Iddo Tzameret. The strength of multilinear proofs. *Computational Complexity*, 17(3):407–457, 2008.
- 29 Robert Reckhow. *On the lengths of proofs in the propositional calculus*. PhD thesis, University of Toronto, 1976. Technical Report No . 87.
- 30 Jacob T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *Journal of the ACM*, 27(4):701–717, 1980.

- 31 Nathan Segerlind. Nearly-exponential size lower bounds for symbolic quantifier elimination algorithms and OBDD-based proofs of unsatisfiability. *Electronic Colloquium on Computational Complexity*, January 2007. ECCC, TR07-009.
- 32 Philip M. Spira. On time-hardware complexity tradeoffs for boolean functions. In *Fourth International Symposium on Systems Sciences*, pages 525–527, 1971.
- 33 Volker Strassen. Vermeidung von divisionen. *J. Reine Angew. Math.*, 264:182–202, 1973. (in German).
- 34 Iddo Tzameret. *Studies in Algebraic and Propositional Proof Complexity*. PhD thesis, Tel Aviv University, 2008.
- 35 Iddo Tzameret. Algebraic proofs over noncommutative formulas. *Information and Computation*, 209(10):1269–1292, 2011.
- 36 Leslie G. Valiant. Completeness classes in algebra. In *Proceedings of the 11th Annual ACM Symposium on the Theory of Computing*, pages 249–261. ACM, 1979.
- 37 Richard Zippel. Probabilistic algorithms for sparse polynomials. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation*, pages 216–226. Springer-Verlag, 1979.

# The Space Complexity of Cutting Planes Refutations

Nicola Galesi<sup>\*1</sup>, Pavel Pudlák<sup>†2</sup>, and Neil Thapen<sup>†2</sup>

- 1 Computer Science Department, Sapienza University of Rome  
via Salaria 113, 00198 Rome, Italy  
nicola.galesi@uniroma1.it
- 2 Institute of Mathematics, Czech Academy of Sciences  
Žitná 25, 115 67 Prague 1, Czech Republic  
{pudlak, thapen}@math.cas.cz

---

## Abstract

We study the space complexity of the *cutting planes* proof system, in which the lines in a proof are integral linear inequalities. We measure the space used by a refutation as the number of linear inequalities that need to be kept on a blackboard while verifying it. We show that any unsatisfiable set of linear inequalities has a cutting planes refutation in space five. This is in contrast to the weaker *resolution* proof system, for which the analogous space measure has been well-studied and many optimal linear lower bounds are known.

Motivated by this result we consider a natural restriction of cutting planes, in which all coefficients have size bounded by a constant. We show that there is a CNF which requires super-constant space to refute in this system. The system nevertheless already has an exponential speed-up over resolution with respect to size, and we additionally show that it is stronger than resolution with respect to space, by constructing constant-space cutting planes proofs, with coefficients bounded by two, of the pigeonhole principle.

We also consider *variable instance space* for cutting planes, where we count the number of instances of variables on the blackboard, and *total space*, where we count the total number of symbols.

**1998 ACM Subject Classification** F.2 Analysis of Algorithms and Problem Complexity, F.2.2. Nonnumerical Algorithms and Problems – Complexity of Proof Procedures

**Keywords and phrases** Proof Complexity, Cutting Planes, Space Complexity

**Digital Object Identifier** 10.4230/LIPIcs.CCC.2015.433

## 1 Introduction

### 1.1 Background

The method of *cutting planes* for integer linear programming was introduced by Gomory [15] and Chvátal [10]. An initial polytope  $P$ , defined by a system of linear inequalities, can be transformed through a sequence of Gomory-Chvátal cuts into the integral hull of  $P$ , that is, into the smallest polytope containing the integral points of  $P$ . If the set of inequalities

---

\* Part of this work was done while visiting the Institute of Mathematics of the Czech Academy of Sciences, partially supported by grant P202/12/G061 of GAČR.

† The research leading to these results has received funding from 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.



defining  $P$  has no integral solution, then the integral hull of  $P$  is empty and the sequence of cuts can be used as a witness that there is no solution.

W. Cook et al. in [12] used this idea to define *cutting-plane* proofs. As we present it in this paper, *cutting planes*, or CP, is a system for refuting unsatisfiable systems of integral linear inequalities over Boolean (0/1-valued) variables. Each line in a CP refutation is an inequality, and there are rules for taking linear combinations and for a version of the Gomory-Chvátal cut (formal definitions follow in Section 2). In particular, CP can be used as a system for refuting unsatisfiable Boolean formulas in conjunctive normal form (CNFs), since these can be translated into sets of inequalities.

Cutting planes has been studied from the point of view of the *size* complexity of proofs, usually measured as the number of lines in a refutation. It has an exponential speed-up over the well-known *resolution* proof system [12]. Exponential lower bounds on size were shown in [17, 24].

By analogy with complexity theory, where we study the space needed by computations, as well as the time, we can also study the space requirements of proofs [14, 1]. In a refutational system based on successively deriving formulas, we imagine presenting a proof by writing formulas on a blackboard as we derive them. We can erase formulas and write down axioms at any time, but if we want to write a formula derived by a rule, all the premises of the rule must be present on the blackboard. How large a blackboard do we need? The most common measure of blackboard size is the number of formulas that will fit on it. This is called in general *formula space*, or *clause space* in resolution or *inequality space* in cutting planes. We also consider some other measures. See Section 2 for definitions.

Space is by now fairly well-understood in resolution (see [22] for a survey) and increasingly also in the algebraic *polynomial calculus* proof system (see e.g. [6]). But little has been known about space in cutting planes. The basic space upper bounds known for resolution [14] carry over to CP, for example, that every unsatisfiable CNF has a refutation with linear space and quadratic total space. W. Cook in [11] showed that every unsatisfiable set of inequalities  $F$  has a refutation with total space polynomial in the space needed to write  $F$  (although his definitions are not quite the same as ours). A nontrivial lower bound for variable instance space in CP is mentioned as an open problem in [1]. Dantchev and Martin in [13] show lower bounds for a certain *width* measure. In a recent paper Göös and Pitassi [16], improving a result of Huynh and Nordström [20], give a family of CNFs of size  $m$  which cannot simultaneously be refuted with small space and small length – the space  $s$  and length  $\ell$  of every CP refutation must satisfy  $s \log \ell \geq m^{1/4-o(1)}$ .

One motivation for studying cutting planes is that it has the potential to offer a more efficient foundation for SAT solving than resolution. From this point of view results about refutation size and refutation space are both interesting, as they may give information about respectively the time and the memory required for computations [23].

## 1.2 Results for cutting planes

Our main result, Theorem 3.4 in Section 3, is a general constant upper bound on the minimal inequality space of CP refutations: any unsatisfiable set of linear inequalities can be refuted in space five. This result, which holds in particular for unsatisfiable CNFs, is in contrast with resolution, where there are several families of CNFs, including random  $k$ -CNFs, which require refutations with linear clause space [14, 3] (the situation is similar with monomial space in polynomial calculus [6]). To prove the theorem we first prove that the *complete tree contradiction*  $CT_n$  has CP refutations in space five (Lemma 3.2), and then use these refutations to build small space refutations for any unsatisfiable set of inequalities.

Section 4 contains three small results that follow from the work in Section 3. First, we observe that the refutations in Lemma 3.2 use coefficients with absolute value at most  $2^n$ . Hence the refutations have total space  $O(n^2)$ , where we measure *total space* by counting the total number of symbols that must be written simultaneously on the blackboard, not just the number of inequalities (we assume that the coefficients are written in binary, and do not consider variable names as taking space – see below). It follows that  $O(n^2)$  total space is sufficient to refute any unsatisfiable set of linear inequalities, as long as the absolute values of the coefficients and the constant term are bounded by an exponential function (Corollary 4.1). Notice however that, restricted to CNFs, this upper bound already follows from the  $O(n^2)$  upper bound for total space in resolution (see e.g. [14, 7]).

Second, we use our derivation of  $\text{CT}_n$  from any unsatisfiable set  $F$  of inequalities to observe, in Proposition 4.2, that  $F$  has a CP refutation in which the absolute values of the coefficients are relatively small – they are bounded by the maximum, over all inequalities  $I$  in  $F$ , of the sum of the absolute values of the coefficients and constant term of  $I$ . This gives smaller bounds than results in [12, 9]; however those are concerned with a different problem, of limiting the size of the coefficients while keeping the refutation short.

Lastly in Section 4 we consider *variable instance space* in CP. This measures the total number of instances of variables that appear simultaneously on the blackboard during a refutation. This is like total space, but ignores the size of the coefficients and constant terms. On the one hand, the minimal width of refuting an unsatisfiable CNF in resolution is a lower bound on the variable instance space in CP; on the other hand, Theorem 3.4 implies a general linear upper bound on variable instance space. This allows us to use known width lower bounds in resolution to show tight linear bounds on variable instance space in CP (Corollary 4.4).

### 1.3 Results for cutting planes with small coefficients

The constant space refutations in Theorem 3.4 use coefficients as big as  $2^n$ , and these seem to be necessary for our proof technique to work. In Sections 5 and 6 we study what can be said about space in CP if we rule out this kind of refutation, by putting an upper bound on the coefficients.

For  $k \in \mathbb{N}$ , we define  $\text{CP}^k$  as the restriction of cutting planes in which every inequality in a derivation must have coefficients with absolute value at most  $k$ . This is already quite a strong proof system for  $k = 2$ . It is exponentially stronger than resolution, since an inspection of the proofs in [12] shows that  $\text{CP}^2$  efficiently simulates resolution and has polynomial size refutations of the pigeonhole principle  $\text{PHP}_m$ . Cutting planes with bounded coefficients has been considered before – the system *generalized resolution* studied in [19] is similar to  $\text{CP}^2$ , and size lower bounds for CP were initially shown for a restricted system  $\text{CP}^*$  with polynomially bounded coefficients [21, 8]. (Note that by a result of [18], if we bound the constant term<sup>1</sup> by  $k$ , rather than bounding the coefficients, we get a system equivalent to resolution.)

In Section 5 we consider a natural candidate for proving inequality space lower bounds on  $\text{CP}^k$  refutations, the pigeonhole principle. We show in Theorem 5.1 that there is no such lower bound for  $\text{PHP}_m$ , and in fact that it has  $\text{CP}^2$  refutations with inequality space five. Our refutation is broadly similar to the refutation in [12] (which uses space linear in the

<sup>1</sup> More precisely, if we write all inequalities in the form  $\sum_{i \in P} \lambda_i x_i + \sum_{i \in N} \lambda_i (1 - x_i) \geq t$  and put a constant upper bound on the term  $t$ .



number of variables). It follows that  $\text{CP}^2$  is strictly stronger than resolution with respect to space.

Finally in Section 6 we prove that small coefficients do not always suffice for constant space proofs, by showing in Theorem 6.6 that for any constant  $k \in \mathbb{N}$ , the contradiction  $\text{CT}_n$  requires inequality space  $\Omega(\log \log \log n)$  to refute in  $\text{CP}^k$ . (In fact we prove something slightly stronger, that the refutation requires many *different* coefficients – our proof does not use the size of the coefficients directly.) Similarly, if we insist on constant inequality space then we get a barely super-constant lower bound on the coefficients. Our lower bounds are very small and surely not optimal. However, the proof is interesting because it is based on a counting argument, which is rare in proof complexity.

The contradiction  $\text{CT}_n$  is unusual in having exponential size in the number  $n$  of variables. However, using a padding argument one can easily show that there is contradiction  $F$  of linear size in  $n$ , and which even has linear size resolution refutations, but which still requires superconstant inequality space to refute in  $\text{CP}^k$  (Corollary 6.7). Nevertheless, it would be interesting to find a more natural example.

## 2 Technical preliminaries

The lines in a cutting planes (CP) proof are inequalities of the form  $\sum \lambda_i x_i \geq t$  where the coefficients  $\lambda_i$  and the constant term  $t$  are integers, and the  $x_i$  are Boolean variables. A CP *derivation* of an inequality  $I$  from a set of inequalities  $F$  is a sequence of lines, ending with  $I$ , where each line is either (1) a member of  $F$ , or (2) a *Boolean axiom*  $x \geq 0$  or  $-x \geq -1$ , or (3) follows from earlier lines by the *linear combination rule* or the *cut rule*. These are respectively

$$\frac{\sum \lambda_i^1 x_i \geq t_1 \quad \cdots \quad \sum \lambda_i^k x_i \geq t_k}{\sum \left( \sum_j s_j \lambda_i^j \right) x_i \geq \sum_j s_j t_j} \quad \text{and} \quad \frac{\sum s \lambda_i x_i \geq t}{\sum \lambda_i x_i \geq \lceil t/s \rceil}$$

where  $s_1, \dots, s_k$  and  $s$  must be strictly positive integers, and the linear combination rule can take any number of premises.<sup>2</sup>

To define our space measures we assume that our derivations come with some extra structure. We follow the model proposed by [14, 1] inspired by the definition of space for Turing machines. A *memory configuration*  $M$  is a set of linear inequalities. A CP *derivation of  $I$  from  $F$*  is then given by a sequence  $M_0, \dots, M_\ell$  of memory configurations, where  $M_i$  represents the contents of the blackboard at the  $i$ th step in the derivation. The sequence must satisfy that  $M_0$  is empty, that  $I \in M_\ell$ , and that for each  $i < \ell$ ,  $M_{i+1}$  is obtained from  $M_i$  in one of three ways:

*Axiom download:*  $M_{i+1} = M_i \cup \{J\}$  for some  $J \in F$

*Inference:*  $M_{i+1} = M_i \cup \{J\}$  where  $J$  follows from  $M_i$  by an inference rule, or is a Boolean axiom

*Erasure:*  $M_{i+1} \subset M_i$ .

<sup>2</sup> One can also define cutting planes using a binary addition rule, a unary multiplication rule and the cut rule. The two systems polynomially simulate each other and the inequality space is exactly the same. However other measures of complexity may differ substantially. We have chosen to use the linear combination rule since this captures better the geometric idea behind cutting planes. We briefly discuss a difference between the systems at the end of Section 4, but other than this our results do not depend essentially on which definition one takes.

A CP refutation of  $F$  is a CP derivation of  $0 \geq 1$  from  $F$ .

We consider three measures of the space taken by a memory configuration  $M$ . The *inequality space* is the number of inequalities in  $M$ . The *variable instance space* is the sum, over all inequalities  $J$  in  $M$ , of the number of distinct variables appearing in  $J$  with a non-zero coefficient (this definition, for general proof systems, is from [1], where it is called simply “variable space”). We define the *total space* as the sum, over all inequalities  $J$  in  $M$ , of the length in binary of all non-zero coefficients in  $J$  and of the constant term of  $J$ , ignoring signs.<sup>3</sup>

For each measure, the corresponding space of a refutation  $\Pi$  is the *maximum* space of any configuration  $M_i$  in  $\Pi$ . The corresponding space needed to refute a set of inequalities  $F$  is the *minimum* space of any refutation of  $F$ . If we refer just to the *space* of a refutation we mean the inequality space, just as in resolution the analogous measure, clause space, is often simply called space.

By an *assignment* to a set of inequalities or CNF  $F$ , we always mean a total assignment of 0/1 values to the variables appearing in  $F$ . We say that  $F$  is unsatisfiable if it is not satisfied by any such assignment.

The *complete tree contradiction*  $\text{CT}_n$ , which is central to this work, is a CNF in  $n$  variables  $x_0, \dots, x_{n-1}$ , with  $2^n$  clauses. For each assignment  $\alpha$ , it contains the clause  $\bigvee_{i \in Z} x_i \vee \bigvee_{i \in A} \neg x_i$  where  $A = \{i : \alpha(x_i) = 1\}$  and  $Z = \{i : \alpha(x_i) = 0\}$ . This clause is falsified by  $\alpha$  and by no other assignment.

We translate propositional clauses into inequalities, and thus CNFs into sets of inequalities, using the translation of [12]:

$$\bigvee_{i \in P} x_i \vee \bigvee_{i \in N} \neg x_i \quad \mapsto \quad \sum_{i \in P} x_i + \sum_{i \in N} (1 - x_i) \geq 1.$$

When describing a CP refutation, we may freely rearrange the terms in an inequality and move the constant term around, for example treating  $\sum \lambda_i x_i \geq t$  and  $\sum \lambda_i x_i + s \geq t + s$  as the same inequality. Similarly, we will sometimes use the Boolean axiom  $-x \geq -1$  in the form  $1 - x \geq 0$ .

When working in a fixed amount of inequality space, it is helpful to think of each unit of space as a “register” that can contain one inequality. We will frequently make use of the following observation, which we record as a lemma:

► **Lemma 2.1.** *If we have one register free, we can treat addition, multiplication and rounding operations as if they happen “in place”, with one of the assumptions overwritten by the conclusion. If we have two registers free, we can add any positive linear combination of axioms to any other register.*

### 3 Inequality space upper bound

We show that any unsatisfiable set of inequalities  $F$  can be refuted in CP in constant inequality space. We do this by first showing that  $\text{CT}_n$  can be refuted in constant space, and then showing that each clause of  $\text{CT}_n$  can be derived from  $F$  in constant space. The overall form of the proof, and the idea of refuting  $\text{CT}_n$  by considering all assignments in

<sup>3</sup> For simplicity, we do not count arithmetical symbols or variable names in total space. Counting these at most trebles the space, if we treat each variable name as a single symbol. It increases it by a factor of  $O(\log n)$  if we include the symbols needed to write variable indices.

lexicographic order, are inspired by the proof of a variable instance space upper bound on the Frege proof system in [1].

We first prove a useful lemma, then the upper bound for  $\text{CT}_n$ .

► **Lemma 3.1.** *Suppose we have two registers free, and a third register that contains an inequality  $\sum_{i \in S} \lambda_i x_i + \sum_{i \in T} \lambda_i (1 - x_i) \geq b$ , with  $b \geq 1$ . Then we can replace the inequality with  $\sum_{i \in S} x_i + \sum_{i \in T} (1 - x_i) \geq 1$ .*

**Proof.** Choose an integer  $c$  greater than or equal to the maximum of  $b$  and all the coefficients  $\lambda_i$ . Using Lemma 2.1, add  $(c - \lambda_i)x_i \geq 0$  to the inequality for each  $i \in S$  and add  $(c - \lambda_i)(1 - x_i) \geq 0$  for each  $i \in T$ . This gives

$$\sum_{i \in S} cx_i + \sum_{i \in T} c(1 - x_i) \geq b.$$

Then divide by  $c$  and round (by applying the cut rule). The constant term becomes  $\lceil b/c \rceil = 1$ . ◀

► **Lemma 3.2.**  *$\text{CT}_n$  has a CP refutation with inequality space 5.*

**Proof.** Given a number  $a < 2^n$  we will write  $(a)_0, \dots, (a)_{n-1}$  for the bits of the binary expansion of  $a$ , so that  $a = \sum 2^i (a)_i$ . Throughout the proof sums  $\sum$  are taken over  $i < n$ , or whichever subset of this is indicated.

For  $a \in \mathbb{N}$ , define the inequality  $T_a$  as

$$T_a : \sum 2^i x_i \geq a.$$

The assignments falsifying  $T_a$  are exactly those lexicographically strictly less than  $a$ . In other words,  $T_a$  is equivalent to the conjunction of the inequalities  $I_b$  over all  $b < a$ , where we write  $I_b$  for the clause of  $\text{CT}_n$  which is falsified exactly by the assignment  $x_i \mapsto (b)_i$ .

For  $a < 2^n$ ,  $T_a$  and  $I_a$  together imply  $T_{a+1}$ . We will show that this implication can be proved in small space. In this way we can proceed by a kind of induction, first deriving  $T_0$ , then deriving in turn  $T_1, T_2, \dots, T_{2^n-1}$  and finally deriving a contradiction from  $T_{2^n-1}$  and  $I_{2^n-1}$ .

For the inductive step, fix  $a < 2^n$ . Let  $A = \{i < n : (a)_i = 1\}$  and  $Z = \{i < n : (a)_i = 0\}$ . Define the inequalities

$$M_a : \sum_{i \in Z} x_i \geq 1 \qquad L_a^k : x_k + \sum_{\substack{i > k \\ i \in Z}} x_i \geq 1.$$

Notice that if  $\beta$  is an assignment such that  $\beta \geq a$  lexicographically, then  $\beta$  satisfies  $L_a^k$  for each  $k \in A$ . If furthermore  $\beta > a$ , then  $\beta$  also satisfies  $M_a$ . We claim these implications are provable in small space:

*Claim 1.* We can derive  $M_a$  from  $T_a$  and  $I_a$  in space 3.

*Claim 2.* We can derive  $L_a^k$  from  $T_a$  in space 3, for any  $k \in A$ .

Using these two claims, we can then show

*Claim 3.* We can derive  $T_{a+1}$  from  $T_a$  and  $I_a$  in space 4.

Here and below by “we can derive” we mean that there exists a CP derivation as defined in Section 2, where we treat the assumptions as axioms which do not take up space until we choose to download them. Using the claims we can carry out the refutation of  $\text{CT}_n$  sketched above, using five registers. The inequality  $T_0$  is a linear combination of the axioms  $x_i \geq 0$  so we may easily derive it in the first register. Then we derive  $T_1$  using  $T_0, I_0$  and the four free

registers, then copy it to the first register. We repeat this for  $T_2, T_3$  and so on. Once we have  $T_{2^n-1}$  we can derive  $M_{2^n-1}$ , which is exactly  $0 \geq 1$ .

It remains to prove the three claims.

*Proof of Claim 1.* We are given  $T_a, I_a$  and three free registers and want to derive  $M_a$ . We write  $I_a$  in the first register, that is,

$$\sum_{i \in Z} x_i + \sum_{i \in A} (1 - x_i) \geq 1.$$

We add to it the following two inequalities, both linear combinations of axioms:

$$\sum_{i \in Z} (2^i - 1)x_i \geq 0 \quad \text{and} \quad \sum_{i \in A} (2^i - 1)(1 - x_i) \geq 0.$$

The result is

$$\sum_{i \in Z} 2^i x_i - \sum_{i \in A} 2^i x_i \geq 1 - \sum_{i \in A} 2^i$$

whose right hand side equals  $1 - a$ . We add  $T_a$  to this, giving

$$2 \sum_{i \in Z} 2^i x_i \geq 1.$$

By Lemma 3.1 we can replace this with  $M_a$ .

*Proof of Claim 2.* We are given  $T_a$  and three free registers and want to derive  $L_a^k$  for a given  $k \in A$ . We copy  $T_a$  into the first register, rearranging it as

$$\sum_{i < k} 2^i x_i + 2^k x_k + \sum_{\substack{i > k \\ i \in A}} 2^i x_i + \sum_{\substack{i > k \\ i \in Z}} 2^i x_i \geq \sum_{\substack{i < k \\ i \in A}} 2^i + 2^k + \sum_{\substack{i > k \\ i \in A}} 2^i.$$

We add the following linear combination of axioms:

$$-\sum_{i < k} 2^i x_i - \sum_{\substack{i > k \\ i \in A}} 2^i x_i \geq -\sum_{i < k} 2^i - \sum_{\substack{i > k \\ i \in A}} 2^i.$$

The result is

$$2^k x_k + \sum_{\substack{i > k \\ i \in Z}} 2^i x_i \geq 2^k - \sum_{i < k} 2^i + \sum_{\substack{i < k \\ i \in A}} 2^i$$

whose right hand side is at least 1. Hence by Lemma 3.1 we can replace it with  $L_a^k$ .

*Proof of Claim 3.* We are given  $T_a, I_a$  and four free registers and want to derive  $T_{a+1}$ . By Claim 1, we can write  $M_a$  in the first register, that is,

$$\sum_{i \in Z} x_i \geq 1.$$

For each  $k \in A$ , we use Claim 2 to write  $L_a^k$  in the second register, and then multiply it by  $2^k$ , giving

$$2^k x_k + 2^k \sum_{\substack{i > k \\ i \in Z}} x_i \geq 2^k.$$

We do this for each  $k \in A$  in turn, each time adding the result to the first register. At the end of this process, the first register contains the inequality

$$\sum_{k \in A} 2^k x_k + \sum_{i \in Z} \left( \sum_{\substack{k < i \\ k \in A}} 2^k \right) x_i + \sum_{i \in Z} x_i \geq 1 + \sum_{k \in A} 2^k.$$

Here the right hand side equals  $a + 1$ , and for  $i \in Z$  the coefficient  $\lambda_i$  of  $x_i$  is less than or equal to  $2^i$ . Hence for all  $i \in Z$  we may add the inequality  $(2^i - \lambda_i)x_i \geq 0$  to the first register, giving

$$\sum_{k \in A} 2^k x_k + \sum_{i \in Z} 2^i x_i \geq a + 1$$

which is  $T_{a+1}$ . ◀

Using the refutation constructed in Lemma 3.2 we prove, in Theorem 3.3, a space upper bound for any unsatisfiable CNF. The proof is simple – any unsatisfiable CNF formula  $F$  in  $n$  variables can be weakened to  $\text{CT}_n$  in resolution (since every assignment falsifies at least one clause) and the Boolean axioms in CP can easily simulate this weakening. We then extend the argument to prove the more general result, an upper bound for any unsatisfiable set of inequalities, as Theorem 3.4.

► **Theorem 3.3.** *Let  $F$  be any unsatisfiable CNF. Then  $F$  has a CP refutation with inequality space 5.*

**Proof.** Suppose  $F$  has variables  $x_0, \dots, x_{n-1}$ . It is enough to show that, for each assignment  $\alpha$ , the inequality  $I_\alpha$  of  $\text{CT}_n$  is derivable in space 4 from the translation of  $F$ . We can then imitate the refutation in the proof of Lemma 3.2.

Let  $\alpha$  be any assignment and let  $A = \{i : \alpha(x_i) = 1\}$  and  $Z = \{i : \alpha(x_i) = 0\}$ . Since  $F$  is unsatisfiable  $\alpha$  falsifies some inequality from  $F$ , of the form

$$I : \sum_{i \in P} x_i + \sum_{i \in N} (1 - x_i) \geq 1.$$

Hence we must have  $\alpha(x_i) = 0$  for each  $i \in P$  and  $\alpha(x_i) = 1$  for each  $i \in N$ . In other words,  $P \subseteq Z$  and  $N \subseteq A$ . Hence we can derive  $I_\alpha$  from  $F$  using space 3, by downloading  $I$  and adding

$$\sum_{i \in Z \setminus P} x_i + \sum_{i \in A \setminus N} (1 - x_i) \geq 0$$

which is a linear combination of axioms. ◀

► **Theorem 3.4.** *Let  $F$  be any set of unsatisfiable inequalities. Then  $F$  has a CP refutation with inequality space 5.*

**Proof.** Suppose  $F$  has variables  $x_0, \dots, x_{n-1}$ . As before, let  $\alpha$  be any assignment and let  $A = \{i : \alpha(x_i) = 1\}$  and  $Z = \{i : \alpha(x_i) = 0\}$ . The assignment  $\alpha$  falsifies some inequality from  $F$ , of the form

$$I : \sum_{i \in P} \lambda_i x_i - \sum_{i \in N} \lambda_i x_i \geq t$$

where  $P$  and  $N$  are disjoint and all the coefficients  $\lambda_i$  are positive. We will derive  $I_\alpha$  from  $I$  in space 3.

We first decompose  $I$  as

$$\sum_{i \in P \cap A} \lambda_i x_i + \sum_{i \in P \cap Z} \lambda_i x_i - \sum_{i \in N \cap A} \lambda_i x_i - \sum_{i \in N \cap Z} \lambda_i x_i \geq t. \quad (1)$$

Since  $I$  is falsified by  $\alpha$ , if we evaluate the left-hand side of (1) under  $\alpha$  we get

$$\sum_{i \in P \cap A} \lambda_i - \sum_{i \in N \cap A} \lambda_i < t.$$

Hence if we set  $\delta = t - \sum_{i \in P \cap A} \lambda_i + \sum_{i \in N \cap A} \lambda_i$  then  $\delta \geq 1$  and we can rewrite (1) as

$$\sum_{i \in P \cap A} \lambda_i x_i + \sum_{i \in P \cap Z} \lambda_i x_i - \sum_{i \in N \cap A} \lambda_i x_i - \sum_{i \in N \cap Z} \lambda_i x_i \geq \sum_{i \in P \cap A} \lambda_i - \sum_{i \in N \cap A} \lambda_i + \delta. \quad (2)$$

We add to (2) the two inequalities

$$-\sum_{i \in P \cap A} \lambda_i x_i \geq -\sum_{i \in P \cap A} \lambda_i \quad \text{and} \quad \sum_{i \in N \cap Z} \lambda_i x_i \geq 0.$$

The result is

$$\sum_{i \in P \cap Z} \lambda_i x_i - \sum_{i \in N \cap A} \lambda_i x_i \geq -\sum_{i \in N \cap A} \lambda_i + \delta$$

which we rearrange as

$$\sum_{i \in P \cap Z} \lambda_i x_i + \sum_{i \in N \cap A} \lambda_i (1 - x_i) \geq \delta.$$

Since  $\delta \geq 1$ , we may use Lemma 3.1 to replace this with

$$\sum_{i \in P \cap Z} x_i + \sum_{i \in N \cap A} (1 - x_i) \geq 1$$

from which we can easily obtain  $I_\alpha$  as in the previous theorem. ◀

## 4 Corollaries

Firstly, from the refutation constructed in Theorem 3.4, we immediately get a general upper bound on the total space needed for CP refutations. Note that there are threshold functions that require coefficients of size  $n^{n/2}$  to write as a linear inequality, so the assumption about the coefficients in  $F$  is necessary.

► **Corollary 4.1.** *Let  $F$  be any unsatisfiable set of linear inequalities over  $n$  variables in which the coefficients and the constant term are bounded by an exponential function  $2^{O(n)}$ . Then  $F$  has a CP refutation with total space  $O(n^2)$  and with coefficients bounded by  $2^{O(n)}$ .*

Secondly, we observe that the reduction to  $\text{CT}_n$  at the end of Section 3 can be used directly to show an upper bound on the size of coefficients needed in a CP refutation.

► **Proposition 4.2.** *Let  $F$  be any set of unsatisfiable inequalities. Let  $\sigma$  be the maximum, over all inequalities  $\sum \lambda_i x_i \geq t$  in  $F$ , of  $\sum |\lambda_i| + |t|$ . Then there exists a CP refutation of  $F$  in which the absolute value of all coefficients is at most  $\sigma$ .*

**Proof.** We use the constructions and notation from the proof of Theorem 3.4. We can derive from  $F$  all inequalities  $I_\alpha$  of  $\text{CT}_n$ . Since these inequalities are translations of clauses, we can then simulate in CP the resolution refutation of  $\text{CT}_n$ . A simulation of resolution uses coefficients with absolute value at most 2. So it remains to check the size of the coefficients in the derivation of each  $I_\alpha$ .

This is derived from a single inequality  $I$  in  $F$ , in two steps. First we obtain an inequality of the form

$$\sum_{i \in P \cap Z} \lambda_i x_i + \sum_{i \in N \cap A} \lambda_i (1 - x_i) \geq \delta \quad (3)$$

where all the  $\lambda_i$  are positive. The coefficients needed to derive this are just the coefficients from  $I$ . Furthermore  $\delta = t - \sum_{i \in P \cap A} \lambda_i + \sum_{i \in N \cap A} \lambda_i$ , so  $|\delta| \leq \sigma$ . We then reduce (3) to

$$\sum_{i \in P \cap Z} x_i + \sum_{i \in N \cap A} (1 - x_i) \geq 1 \quad (4)$$

as in Lemma 3.1, by letting  $c = \max\{\lambda_1, \dots, \lambda_n, \delta\}$ , adding  $(c - \lambda_i)x_i \geq 0$  to (3) for each  $i \in P \cap Z$ , adding  $(c - \lambda_i)(1 - x_i) \geq 0$  to (3) for each  $i \in N \cap A$ , and then dividing by  $c$  and rounding. Since  $c$  and all the  $\lambda_i$  are positive, the largest coefficient that appears in this process is at most  $\max\{|\lambda_1|, \dots, |\lambda_n|, c\}$ , which is bounded by  $\sigma$ .

From (4) we can get  $I_\alpha$  using only coefficients  $\pm 1$ . In fact, we do not even need this step, since (4) already is the translation of a clause, and the collection of all such clauses has a resolution refutation.  $\blacktriangleleft$

Lastly we briefly discuss bounds on variable instance space in CP. The *width* of a resolution refutation is the size of the largest clause in it. We state the next lemma for variable instance space, but we will show a stronger fact that resolution width is at most the “variable space without repetitions” of refuting  $F$  in CP, where the space of a configuration is measured by counting the number of different variables that appear (this measure is called simply “variable space” in [4, 2]). It is well-known that this fact is true for any refutation system based on formulas defining Boolean functions (see Lemma 8 of [2]).

► **Lemma 4.3.** *Let  $F$  be an unsatisfiable CNF. The minimal width of refuting  $F$  in resolution is at most the variable instance space of refuting  $F$  in CP.*

**Proof.** Let  $\Pi$  be a CP refutation of  $F$  in which every configuration contains at most  $s$  many different variables with non-zero coefficients. We sketch how to simulate  $\Pi$  by a resolution refutation  $\rho$  with width at most  $s$ . For any inequality  $I$  in  $\Pi$ , let  $X$  be the set of variables in  $I$  with non-zero coefficients, and let  $\Phi_I$  be a CNF in variables  $X$  expressing the same Boolean function as  $I$ . Let  $I_1, \dots, I_m$  be the inequalities from which  $I$  was derived by a rule in  $\Pi$ . Then there is a resolution derivation of  $\Phi_I$  from  $\Phi_{I_1}, \dots, \Phi_{I_m}$ , since resolution is implicationally complete. The total number of different variables appearing in this derivation is at most  $s$ , since  $I_1, \dots, I_m$  and  $I$  must belong to the same configuration in  $\Pi$ , hence can mention no more than  $s$  variables in total. In particular, the width of the resolution derivation is at most  $s$ .  $\blacktriangleleft$

The lemma allows us to use known lower bounds on width in resolution, together with the linear upper bound on variable instance space that follows immediately from Theorem 3.4, to derive tight bounds on variable instance space in CP. For example, using a result of [5], we get:



► **Corollary 4.4.** *With high probability the variable instance space of refuting a random  $k$ -CNF in CP is  $\Theta(n)$ .*

Note that if we had defined cutting planes using a binary addition rule and unary multiplication rule (rather than arbitrary linear combinations), the simulation in Lemma 4.3 would prove that resolution width is at most twice the CP width, if we define the width of an inequality as the number of variables appearing with non-zero coefficients. Clearly, in such a proof the particular form of the rules used is irrelevant; only their arity matters. In the version of CP we use, it is not so easy to prove non-trivial width lower bounds. Dantchev and Martin in [13] show a width lower bound for an ordering principle in essentially our system, using a geometrical argument.

## 5 PHP<sub>n</sub> with small coefficients

We consider the pigeonhole principle contradiction PHP<sub>n</sub>. It is formalized, as usual, by the following set of inconsistent inequalities:

$$P_i : \sum_{j < n} x_{ij} \geq 1 \quad \text{for } i < n + 1$$

$$H_{ii'j} : x_{ij} + x_{i'j} \leq 1 \quad \text{for } i < i' < n + 1 \text{ and } j < n.$$

To simplify our presentation we will be less strict about how we write inequalities in CP refutations, and allow the notation  $\sum \lambda_i x_i \leq t$  (we do not change the formal rules of the system). With this notation the Boolean axioms look like  $-x \leq 0$  and  $x \leq 1$  and the cut rule looks like

$$\frac{\sum s \lambda_i x_i \leq t}{\sum \lambda_i x_i \leq \lfloor t/s \rfloor}$$

where we round the constant term down rather than up.

► **Theorem 5.1.** *PHP<sub>n</sub> has polynomial size CP<sup>2</sup> refutations with space 5.*

The non-trivial part of the proof is taken care of by the following lemma.

► **Lemma 5.2.** *Given inequalities  $y_i + y_j \leq 1$  for all  $i < j < n$ , we can derive  $\sum y_i \leq 1$  in polynomial size and in space 4, using coefficients bounded by 2.*

**Proof.** Let  $A_m$  be the inequality

$$A_m : \sum_{i < m} y_i \leq 1.$$

We claim that, for  $m < n$ ,  $A_{m+1}$  can be derived from  $A_m$  in space 3. The lemma follows immediately.

So suppose we are given  $A_m$ , all inequalities  $y_i + y_j \leq 1$ , and three free registers. Our strategy is to derive the inequality

$$B_k : \sum_{i < k} y_i + y_m \leq 1 \tag{5}$$

in the first register, for  $k = 1, \dots, m - 1$  in turn. For  $k = 1$  this is an axiom, and for  $k = m - 1$  it is  $A_{m+1}$ , as required. Suppose we have derived  $B_k$  for some  $1 \leq k < m - 1$  and want to derive  $B_{k+1}$ . We add to  $B_k$  the inequalities

$$y_k + y_m \leq 1 \quad \text{and} \quad \sum_{i < k+1} y_i \leq 1. \tag{6}$$

The first of these is an axiom. The second is a weakening of  $A_m$ , which we could derive in three registers by downloading  $A_m$  and then adding the combination of Boolean axioms  $-y_{k+1} - \dots - y_{m-1} \leq 0$ . However, since we only have two registers free, we achieve the same effect by adding  $A_m$  and  $-y_{k+1} - \dots - y_{m-1} \leq 0$  directly to the first register. The result is

$$\sum_{i < k} 2y_i + 2y_k + 2y_m \leq 3$$

since each index appears exactly twice in the three inequalities from (5) and (6). We derive  $B_{k+1}$  by dividing by two and rounding down the constant term. ◀

**Proof of Theorem 5.1.** We are given the  $\text{PHP}_n$  axioms and five free registers. We use Lemma 5.2 and the first four registers to derive

$$\sum_{i < n+1} x_{ij} \leq 1$$

for each  $j < n$  in turn, each time adding the result to the fifth register. The fifth register then contains the total

$$\sum_{j < n} \sum_{i < n+1} x_{ij} \leq n, \quad \text{or equivalently} \quad - \sum_{i < n+1} \sum_{j < n} x_{ij} \geq -n.$$

We obtain  $0 \geq 1$  by adding to this the axioms  $P_i$  for all  $i < n + 1$ . ◀

## 6 Space lower bounds for small coefficients

We use a counting argument to show that any CP refutation of  $\text{CT}_n$ , in which there is a global constant bound on the number of different coefficients appearing in every configuration, must have superconstant inequality space. In particular, this implies superconstant lower bounds on inequality space for  $\text{CT}_n$  in the system  $\text{CP}^k$ .

► **Definition 6.1.** Call a set  $A$  of assignments *s-symmetric* if there is a partition of the variables into  $s$  or fewer blocks, such that  $A$  is closed under every permutation which preserves all blocks.

► **Lemma 6.2.** *Suppose  $I$  is a linear inequality in which no more than  $b$  different coefficients appear. Then the set of assignments falsifying  $I$  is  $b$ -symmetric.*

*Suppose  $M$  is a CP configuration in space  $c$ , such that no more than  $b$  different coefficients appear in any inequality in  $M$ . Then the set of assignments falsifying  $M$  is  $b^c$ -symmetric.*

**Proof.** For the first part, the inequality  $I$  has the form

$$\lambda_1 \sum_{i \in B_1} x_i + \dots + \lambda_b \sum_{i \in B_b} x_i \geq t.$$

The  $b$ -symmetry is witnessed by the blocks  $B_1, \dots, B_b$ . For the second part, take the common refinement of the partitions for all of the inequalities in  $M$ . ◀

► **Lemma 6.3.** *Suppose that  $\text{CT}_n$  has a CP refutation in space  $c$ , in which no more than  $b$  different coefficients appear in any inequality. Then there is a sequence  $A_1, \dots, A_N$  of sets of  $b^c$ -symmetric assignments, beginning with the empty set and ending with the set of all assignments, such that for each  $i < N$  either  $A_{i+1} \subseteq A_i$  or  $A_{i+1} = A_i \cup \{\alpha\}$  for some assignment  $\alpha$ .*

**Proof.** Let  $A_i$  be the set of assignments falsifying the  $i$ th configuration. ◀

We define a  $k$ -assignment to be an assignment with exactly  $k$  variables set to 1 and all the rest set to 0.

► **Lemma 6.4.** Define  $S(s, k) = \{|A| : A \text{ is an } s\text{-symmetric set of } k\text{-assignments}\}$ . Then  $|S(s, k)| < n^s 2^{k^s}$ .

This is proved after Theorem 6.5.

► **Theorem 6.5.** For  $n \geq 2$ , suppose that  $\text{CT}_n$  has a CP refutation in space  $c$ , in which no more than  $b$  different coefficients appear in any inequality. Then  $b^c \geq \sqrt{\log \log n}$ .

**Proof.** Let  $s = b^c$  and  $k = 2^s$ . For trivial reasons  $b, c \geq 2$  so  $s \geq 4$ .

Let  $A_1, \dots, A_N$  be the sequence of  $s$ -symmetric assignments from Lemma 6.3, and let  $A'_i = \{\alpha \in A_i : \alpha \text{ is a } k\text{-assignment}\}$ . Then  $A'_1$  is empty,  $A'_N$  consists of all  $k$ -assignments, and for each  $i < N$  either  $A'_{i+1} \subseteq A'_i$  or  $A'_{i+1} = A'_i \cup \{\alpha\}$  for some  $k$ -assignment  $\alpha$ . It follows that the sequence  $|A'_1|, \dots, |A'_N|$  must contain every number between 0 and  $\binom{n}{k}$ . Since each  $A'_i$  is still  $s$ -symmetric, this in particular means that for every number  $m$  between 0 and  $\binom{n}{k}$ , there is at least one  $s$ -symmetric set  $A$  of  $k$ -assignments with  $|A| = m$ .

Hence, in the notation of Lemma 6.4,  $S(s, k) = \binom{n}{k} + 1$ . It follows that  $\binom{n}{k} < n^s 2^{k^s}$ . Using the bound  $\left(\frac{n}{k}\right)^k \leq \binom{n}{k}$  and taking the logarithm of both sides, we get

$$k(\log n - \log k) < s \log n + k^s.$$

Substituting  $k = 2^s$  gives

$$2^s(\log n - s) < s \log n + 2^{s^2}.$$

Now assume for a contradiction that  $s < \sqrt{\log \log n}$ . Then  $\log n - s \geq \frac{1}{2} \log n$  (we may assume  $n \geq 4$ ) and  $2^{s^2} < \log n$ . The inequality becomes

$$2^{s-1} \log n < (s + 1) \log n$$

which is impossible. ◀

**Proof of Lemma 6.4.** Let  $A$  be an  $s$ -symmetric set of  $k$ -assignments. Let  $B_1, \dots, B_s$  be a partition witnessing the  $s$ -symmetry (we allow some of the blocks to be empty). Then  $A$  is the union of orbits, where each orbit is parametrized by a distinct tuple  $r_1, \dots, r_s$  summing to  $k$ , and the orbit consists of every  $k$ -assignment which has exactly  $r_i$  many ones in each block  $B_i$ . Let  $n_i = |B_i|$ . Then

$$|A| = \sum_{j=1}^m \binom{n_1}{r_1^j} \cdot \dots \cdot \binom{n_s}{r_s^j}$$

where there are  $m$  orbits and the  $j$ th orbit has parameters  $\bar{r}^j = r_1^j, \dots, r_s^j$ . In particular,  $|A|$  depends only on the sizes  $n_1, \dots, n_s$  and on the set of tuples  $\{\bar{r}^1, \dots, \bar{r}^m\}$  characterizing the set of orbits. There are no more than  $n^s$  ways to choose  $n_1, \dots, n_s$ . There are no more than  $k^s$  ways to choose the parameters  $\bar{r}$  for an orbit, and therefore there are no more than  $2^{k^s}$  possible sets of such parameters. Therefore there are at most  $n^s 2^{k^s}$  possible values for  $|A|$ . ◀

From Theorem 6.5 we immediately get:

► **Theorem 6.6.** For any constant  $k \in \mathbb{N}$ , the complete tree contradiction  $\text{CT}_n$  requires inequality space  $\Omega(\log \log \log n)$  to refute in  $\text{CP}^k$ .

► **Corollary 6.7.** There is a family of propositional CNFs  $F$  in  $n$  variables, with linear size and with linear sized resolution refutations, which require superconstant inequality space to refute in  $\text{CP}^k$  for any fixed  $k \in \mathbb{N}$ .

**Proof.** Let  $m = \log n$ , and let  $F$  be  $\text{CT}_m$  together with  $2^m - m$  inequalities of the form  $y_i \geq 1$  in variables  $y_1, \dots, y_{2^m - m}$  disjoint from the variables in  $\text{CT}_m$ . Then  $F$  has a resolution refutation of linear size, since  $\text{CT}_m$  has a refutation of size  $2^m$ , and any constant-space  $\text{CP}^2$  refutation of  $F$  can be made into a constant-space  $\text{CP}^2$  refutation of  $\text{CT}_m$  by substituting 1 for all variables  $y_i$ . ◀

We note that, as in Section 4, our lower bound relies only on the class of Boolean functions appearing as lines in the refutation, not on the particular rules used.

## 7 Open problems

There are many problems about cutting planes that are worth mentioning, but we confine ourselves to a small sample, directly connected with the results presented in this paper.

The first general problem is about the trade-off between inequality space and the size of coefficients. Our upper bound uses coefficients of exponential size, while we can only prove that if space is constant then coefficients can be lower-bounded by a very slowly growing function. In particular the following is open:

*Problem 1.* Can every unsatisfiable CNF be refuted in CP in constant space, if the coefficients are polynomially bounded?

A related open problem is:

*Problem 2.* Can every unsatisfiable CNF be refuted in CP in linear total space?

It seems plausible that some extension of the proof of Theorem 6.6 might work also for such a lower bound.

Among the restricted systems of CP, the system  $\text{CP}^2$  stands out as already being strong enough to simulate resolution and to capture some of the counting available in CP, since it has efficient proofs of  $\text{PHP}_n$ . It would be interesting to improve our results at least for this system. In particular:

*Problem 3.* Prove a better space lower bound for  $\text{CP}^2$ .

---

## References

- 1 Michael Alekhovich, Eli Ben-Sasson, Alexander A Razborov, and Avi Wigderson. Space complexity in propositional calculus. *SIAM Journal on Computing*, 31(4):1184–1211, 2002.
- 2 Chris Beck, Jakob Nordstrom, and Bangsheng Tang. Some trade-off results for polynomial calculus. In *Proceedings of the 45th Annual Symposium on the Theory of Computing (STOC 2013)*, pages 813–822, 2013.
- 3 Eli Ben-Sasson and Nicola Galesi. Space complexity of random formulae in resolution. *Random Structures and Algorithms*, 23(1):92–109, 2003.
- 4 Eli Ben-Sasson and Jakob Nordström. Understanding space in proof complexity: separations and trade-offs via substitutions. In *Proceedings of the 2nd Symposium on Innovations in Computer Science (ICS '11)*, pages 401–416, 2011.

- 5 Eli Ben-Sasson and Avi Wigderson. Short proofs are narrow – resolution made simple. *Journal of the ACM*, 48(2):149–169, 2001.
- 6 Ilario Bonacina and Nicola Galesi. Pseudo-partitions, transversality and locality: a combinatorial characterization for the space measure in algebraic proof systems. In *Proceedings of the 4th conference on Innovations in Theoretical Computer Science (ITCS '13)*, pages 455–472, 2013.
- 7 Ilario Bonacina, Nicola Galesi, and Neil Thapen. Total space in resolution. In *55th IEEE Annual Symposium on Foundations of Computer Science (FOCS 2014)*, pages 641–650, 2014.
- 8 Maria Bonet, Toniann Pitassi, and Ran Raz. Lower bounds for cutting planes proofs with small coefficients. *Journal of Symbolic Logic*, 62(03):708–728, 1997.
- 9 Samuel R Buss and Peter Clote. Cutting planes, connectivity, and threshold logic. *Archive for Mathematical Logic*, 35(1):33–62, 1996.
- 10 Vašek Chvátal. Edmonds polytopes and a hierarchy of combinatorial problems. *Discrete Mathematics*, 4(4):305–337, 1973.
- 11 William Cook. Cutting-plane proofs in polynomial space. *Mathematical Programming*, 47(1-3):11–18, 1990.
- 12 William Cook, Collette R Coullard, and György Turán. On the complexity of cutting-plane proofs. *Discrete Applied Mathematics*, 18(1):25–38, 1987.
- 13 Stefan Dantchev and Barnaby Martin. Cutting planes and the parameter cutwidth. *Theory of Computing Systems*, 51(1):50–64, 2012.
- 14 Juan Luis Esteban and Jacobo Torán. Space bounds for resolution. *Information and Computation*, 171(1):84–97, 2001.
- 15 Ralph E Gomory. An algorithm for integer solutions to linear programs. *Recent Advances in Mathematical Programming*, 64:260–302, 1963.
- 16 Mika Göös and Toniann Pitassi. Communication lower bounds via critical block sensitivity. In *Proceedings of the 46th Annual Symposium on the Theory of Computing (STOC 2014)*, pages 847–856, 2014.
- 17 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.
- 18 Edward A Hirsch and Sergey I Nikolenko. Simulating cutting plane proofs with restricted degree of falsity by resolution. In *Proceedings of the 8th International Conference on Theory and Applications of Satisfiability Testing (SAT 2005)*, pages 135–142, 2005.
- 19 John N Hooker. Generalized resolution for 0–1 linear inequalities. *Annals of Mathematics and Artificial Intelligence*, 6(1-3):271–286, 1992.
- 20 Trinh Huynh and Jakob Nordström. On the virtue of succinct proofs: amplifying communication complexity hardness to time-space trade-offs in proof complexity. In *Proceedings of the 44th Annual Symposium on the Theory of Computing (STOC 2012)*, pages 233–248, 2012.
- 21 Russell Impagliazzo, Toniann Pitassi, and Alasdair Urquhart. Upper and lower bounds for tree-like cutting planes proofs. In *Proceedings of the 9th Annual IEEE Symposium on Logic in Computer Science (LICS '94)*, pages 220–228, 1994.
- 22 Jakob Nordström. Pebble games, proof complexity, and time-space trade-offs. *Logical Methods in Computer Science*, 9(3):15, 2013.
- 23 Jakob Nordström. A (biased) proof complexity survey for SAT practitioners. In *Proceedings of the 17th International Conference on Theory and Applications of Satisfiability Testing (SAT 2014)*, volume 8561, pages 1–6, 2014.
- 24 Pavel Pudlák. Lower bounds for resolution and cutting plane proofs and monotone computations. *Journal of Symbolic Logic*, 62(03):981–998, 1997.

# Tight Size-Degree Bounds for Sums-of-Squares Proofs

Massimo Lauria and Jakob Nordström

KTH Royal Institute of Technology  
SE-100 44 Stockholm, Sweden

---

## Abstract

---

We exhibit families of 4-CNF formulas over  $n$  variables that have sums-of-squares (SOS) proofs of unsatisfiability of degree (a.k.a. rank)  $d$  but require SOS proofs of size  $n^{\Omega(d)}$  for values of  $d = d(n)$  from constant all the way up to  $n^\delta$  for some universal constant  $\delta$ . This shows that the  $n^{O(d)}$  running time obtained by using the Lasserre semidefinite programming relaxations to find degree- $d$  SOS proofs is optimal up to constant factors in the exponent. We establish this result by combining NP-reductions expressible as low-degree SOS derivations with the idea of relativizing CNF formulas in [Krajíček '04] and [Dantchev and Riis '03], and then applying a restriction argument as in [Atserias, Müller, and Oliva '13] and [Atserias, Lauria, and Nordström '14]. This yields a generic method of amplifying SOS degree lower bounds to size lower bounds, and also generalizes the approach in [ALN14] to obtain size lower bounds for the proof systems resolution, polynomial calculus, and Sherali-Adams from lower bounds on width, degree, and rank, respectively.

**1998 ACM Subject Classification** F.2.3 [Analysis of Algorithms and Problem Complexity] Trade-offs among Complexity Measures, F.1.3 [Computation by Abstract Devices] Complexity Measures and Classes – Relations among complexity measures, I.2.3 [Artificial Intelligence] Deduction and Theorem Proving, F.4.1 [Mathematical Logic and Formal Languages] Mathematical Logic – computational logic

**Keywords and phrases** Proof complexity, resolution, Lasserre, Positivstellensatz, SOS, sums-of-squares, semidefinite programming, size, degree, rank, clique, lower bound

**Digital Object Identifier** 10.4230/LIPIcs.CCC.2015.448

## 1 Introduction

Let  $f_1, \dots, f_s \in \mathbb{R}[x_1, \dots, x_n]$  be real, multivariate polynomials. Then the Positivstellensatz proven in [20, 31] says (as a special case) that the the system of equations

$$f_1 = 0, \dots, f_s = 0 \tag{1.1}$$

has no solution over  $\mathbb{R}^n$  if and only if there exist polynomials  $g_j, q_\ell \in \mathbb{R}[x_1, \dots, x_n]$  such that

$$\sum_{j=1}^s g_j f_j = -1 - \sum_{\ell} q_\ell^2 . \tag{1.2}$$

That there can exist no solution given an expression of the form (1.2) is clear, but what is more interesting is that there always exists such an expression to certify unsatisfiability. We refer to (1.2) as a *Positivstellensatz proof* or *Sums-of-squares (SOS) proof* of unsatisfiability,



© Massimo Lauria and Jakob Nordström;  
licensed under Creative Commons License CC-BY  
30th Conference on Computational Complexity (CCC'15).  
Editor: David Zuckerman; pp. 448–466



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

or as an *SOS refutation*,<sup>1</sup> of (1.1). We remark that the Positivstellensatz also applies if we add inequalities  $h_1 \geq 0, \dots, h_t \geq 0$  to the system of equations and allow terms  $-h_j \sum_{\ell} q_{j,\ell}^2$  on the right-hand side in (1.2).

The *degree*<sup>2</sup> of an SOS refutation is the maximal degree of any  $g_j f_j$ . The search for proofs of constant degree  $d$  is *automatizable* as shown in a sequence of works by Shor [30], Nesterov [25], Lasserre [21], and Parrilo [27]. What this means is that if there exists a degree- $d$  SOS refutation for a system of polynomial equalities (and inequalities) over  $n$  variables, then such a refutation can be found in polynomial time  $n^{O(d)}$ . Briefly, one can view (1.2) as linear system of equations in the coefficients of  $g_j$  and  $u = \sum_{\ell} q_{\ell}^2$  with the added constraint that  $u$  is a sum of squares, and such a system can be solved by semidefinite programming in  $d/2$  rounds of the Lasserre SDP hierarchy.

In the last few years there has been renewed interest in sums-of-squares in the context of constraint satisfaction problems (CSPs) and hardness of approximation, as witnessed by, for instance, [3, 26, 32]. These works have highlighted the importance of SOS degree upper bounds for CSP approximability, and this is currently a very active area of study.

Our focus in this paper is not on algorithmic questions, however, but more on sums-of-squares viewed as a proof system (also referred to in the literature as *Positivstellensatz* or *Lasserre*). This proof system was introduced by Grigoriev and Vorobjov [15] as an extension of the Nullstellensatz proof system studied by Beame et al. [5], and Grigoriev established SOS degree lower bound for unsatisfiable  $\mathbb{F}_2$ -linear equations [13] (also referred to as the 3-XOR problem when each equation involves at most 3 variables) and for the knapsack problem [12].

Given the connections to semidefinite programming and the Lasserre SDP hierarchy, it is perhaps not surprising that most works on SOS lower bounds have focused on the degree measure. However, from a proof complexity point of view it is also natural to ask about the minimal *size* of SOS proofs, measured as the number of monomials when all polynomials in each term in (1.2) are expanded out as linear combinations of monomials. Such SOS size lower bounds were proven for knapsack in [14] and  $\mathbb{F}_2$ -linear systems of equations in [18],<sup>3</sup> and tree-like size lower bounds for other formulas were also obtained in [28].

A wider interest in this area of research was awakened when Schoenebeck [29] (essentially) rediscovered Grigoriev’s result [13], which together with further work by Tulsiani [32] led to integrality gaps for a number of constraint satisfaction problems. There have also been papers such as [6] and [16] focusing on *semantic* versions of the proof system, with less attention to the actual syntactic derivation rules used. We refer the reader to, for instance, the introductory section of [26] for more background on sums-of-squares and connections to hardness of approximation, and to the survey [4] for an in-depth discussion of SOS as an approximation algorithm and the intriguing connections to the so-called Unique Games Conjecture [17].

## 1.1 Our Contribution

As discussed above, if a system of polynomial equalities and inequalities over  $n$  variables can be shown inconsistent by SOS in degree  $d$ , then by using semidefinite programming one

<sup>1</sup> All proofs for systems of polynomial equations or for formulas in conjunctive normal form (CNF) in this paper will be proofs of unsatisfiability, and we will therefore use the two terms “proof” and “refutation” interchangeably.

<sup>2</sup> This is sometimes also referred to as the “rank,” but we will stick to the term “degree” in this paper.

<sup>3</sup> It might be worth pointing out that definitions and terminology in this area have suffered from a certain lack of standardization, and so what [18] refers to as “static Lovász-Schrijver calculus” is closer to what we mean by SOS/Lasserre.



can find an SOS refutation of the system in time  $n^{O(d)}$ . It is natural to ask whether this is optimal, or whether there might exist “shortcuts” that could lead to SOS refutations more quickly.

We prove that there are no such shortcuts in general, but that the running time obtained by using the Lasserre semidefinite programming relaxations to find SOS proofs is optimal up to the constant in the exponent. We show this by constructing formulas on  $n$  variables (which can be translated to systems of polynomial equalities in a canonical way) that have SOS refutations of degree  $d$  but require refutations of size  $n^{\Omega(d)}$ . Our lower bound proof works for  $d$  from constant all the way up to  $n^\delta$  for some constant  $\delta$ .

► **Theorem 1.1 (informal).** *Let  $d = d(n) \leq n^\delta$  where  $\delta > 0$  is a universal constant. Then there is a family of 4-CNF formulas  $\{F_n\}_{n \in \mathbb{N}^+}$  with  $O(n^2)$  clauses over  $O(n)$  variables such that  $F_n$  is refutable in sums-of-squares in degree  $\Theta(d)$  but any SOS refutation of  $F_n$  requires size  $n^{\Omega(d)}$ .*

This theorem extends an analogous result joint by the two authors with Atserias in [1] for the proof systems resolution, polynomial calculus, and Sherali-Adams,<sup>4</sup> where upper bounds on refutation size in terms of width, degree, and rank, respectively, were shown to be tight up to the multiplicative constant in the exponent. Theorem 1.1 works for all of these proof systems, since the upper bound is in fact on resolution width (i.e., the size of a largest clause in a resolution refutation), not just SOS degree, and in this sense the theorem subsumes the results in [1]. The concrete bound we obtain for the exponent inside the asymptotic notation in the  $n^{\Omega(d)}$  size lower bound is very much worse, however, and therefore the gap between upper and lower bounds is very much larger than in [1].

We want to emphasize that the size lower bound in Theorem 1.1 holds for SOS proofs of arbitrary degree. Thus, going to higher degree (i.e., higher levels of the Lasserre SDP hierarchy) does not help, since even arbitrarily large degree cannot yield shorter proofs. This is an interesting parallel to the paper [24] exhibiting problems for which a (symmetric) SDP relaxation of arbitrary degree but bounded size  $n^d$  does not do much better than the systematic relaxation of degree  $d$ .

## 1.2 Techniques

We obtain the result in Theorem 1.1 as a special case of a more general method of amplifying lower bounds on width (in resolution), degree (in polynomial calculus) and rank/degree (in Sherali-Adams and Lasserre/SOS) to size lower bounds in the corresponding proof systems. This method is in some sense already implicit in [1], which in turn relies heavily on an earlier paper by Atserias et al. [2], but it turns out that extracting the essential ingredients and making them explicit is helpful for extending the results in [1] to an analogue for sums-of-squares. We give a brief, informal description of the three main ingredients of the method below.

**(i) Find a base CNF formulas hard with respect to width/degree/rank.** To start, we need to find a base problem, encoded as an unsatisfiable CNF formula, that is “moderately hard” for the proof system at hand. What this means is that we should be able to prove asymptotically tight bounds on width if we are dealing with resolution, on degree for polynomial calculus,

---

<sup>4</sup> The exact details of these proof systems are not important for this discussion, and so we choose not to elaborate further here, instead referring the interested reader to [1].

and on degree/rank for Sherali-Adams and sums-of-squares. It then follows by a generic argument (as discussed briefly above for SOS) that a bound  $O(d)$  on width/degree/rank implies an upper bound  $n^{O(d)}$  on proof size.

In [1, 2] the pigeonhole principle served as the base problem. This principle, which has been extensively studied in proof complexity, is encoded in CNF as *pigeonhole principle (PHP) formulas* saying that there is a one-to-one mapping of  $m$  pigeons into  $n$  pigeonholes for  $m > n$ . For sums-of-squares we cannot use PHP formulas, however, since they are not hard with respect to SOS degree. Instead we construct an SOS reduction in low degree from inconsistent systems of  $\mathbb{F}_2$ -linear equations to the clique problem, and then appeal to the result in [13, 29] briefly discussed above to obtain the following degree lower bound.

► **Theorem 1.2 (informal).** *Given  $k \in \mathbb{N}^+$ , there is a graph  $G$  and a 3-CNF formula  $k$ -Clique( $G$ ) of size polynomial in  $k$  with the following properties:*

1. *The graph  $G$  does not contain a  $k$ -clique, but the formula  $k$ -Clique( $G$ ) claims that it does.*
2. *Resolution can refute  $k$ -Clique( $G$ ) in width  $k$ .*
3. *Any sums-of-squares refutation of  $k$ -Clique( $G$ ) requires degree  $\Omega(k)$ .*

(ii) **Relativize the CNF formulas.** The second step is to take the formulas for which we have established width/degree/rank lower bounds and *relativize* them. Relativization is an idea that seems to have been considered for the first time in the context of proof complexity by Krajíček [19] and that was further developed by Dantchev and Riis [11]. Very loosely, it can be described as follows.

Suppose that we have a CNF formula encoding (the negation of) a combinatorial principle saying that some set  $S$  has a property. For instance, the CNF formula could encode the pigeonhole principle discussed above, or could claim the existence of a totally ordered set of  $n$  elements where no element in the set is minimal with respect to the ordering (these latter CNF formulas are known as *ordering principle formulas*, *least number principle formulas*, or *graph tautologies* in the literature).

The formula at hand is then relativized by constructing another formula encoding that there is a (potentially much larger) set  $T$  containing a subset  $S \subseteq T$  for which the same combinatorial principle holds. For the ordering principle, we can encode that there exists a non-empty ordered subset  $S \subseteq T$  of arbitrary size such that it is possible for all elements in  $S$  to find a smaller element inside  $S$ . This relativization step transforms the previously very easy ordering principle formulas into relativized versions that are exponentially hard for resolution [9, 10]. For the PHP formulas, we specify that we have a set of  $M \gg m$  pigeons mapped into  $n < m$  holes such that there exists a subset of  $m$  pigeons that are mapped injectively.

In our setting, it will be important that the relativization does not make the formulas too hard. We do not want the hardness to blow up exponentially and instead would like the upper bound obtained in the first step above to scale nicely with the size of the relativization. For our general approach to work, we therefore need formulas talking about some domain being mapped to some range, where we can enlarge the domain while keeping the range fixed, and where in addition the mapping is symmetric in the sense that permuting the domain does not change the formula.

For this reason, relativizing the ordering principle formulas does not work for our purposes. Pigeonhole principle formulas have this structure, however, which is exactly why the proofs in [1] go through. As already mentioned, PHP formulas will not work for sums-of-squares, but we can relativize the formulas in Theorem 1.2 by saying that there is a large subset of vertices such that there is a  $k$ -clique hiding inside such a subset.

**(iii) Apply random restrictions to show proof size lower bounds.** In the final step, we use random restrictions to establish lower bounds on proof size for the relativized CNF formulas obtained in the second step. This part of the proof is relatively standard, except for a crucial twist in the restriction argument introduced in [2].

Assume that there is a small refutation in sums-of-squares (or whatever proof system we are studying) of the relativized formula claiming the existence of a subset of size  $m \ll M$  with the given combinatorial property. Now hit the formula (and the refutation) with a random restriction that in effect chooses a subset of size  $m$ , and hence gives us back the original, non-relativized formula. This restriction will be fairly aggressive in terms of the number of variables set to fixed truth values, and hence it will hold with high probability that the restricted refutation has no monomials of high degree (or, for resolution, no clauses of high width), since all such monomials will either have been killed by the restriction or at least have shrunk significantly. (We remark that making use of this shrinking in the analysis is the crucial extra feature added in [2].) But this means that we have a refutation of the original formula in degree smaller than the lower bound established in the first step. Hence, no small refutation can exist, and the lower bound on proof size follows.

This concludes the overview of our method to amplify lower bounds on width/degree/rank to size. It is our hope that developing such a systematic approach for deriving this kind of lower bounds, and making explicit what conditions are needed for this approach to work, can also be useful in other contexts.

### 1.3 Organization of This Paper

The rest of this paper is organized as follows. We start in Section 2 by reviewing the definitions and notation used, and also stating some basic facts that we will need. In Section 3, we prove a degree lower bound for CNF formulas encoding a version of the clique problem. We then present in Section 4 a general method for obtaining SOS size lower bounds from degree lower bounds (or from width, degree, and rank, respectively, for proof systems such as resolution, polynomial calculus, and Sherali-Adams). We conclude with a brief discussion of some possible directions for future research in Section 5. We refer to the full-length version [22] of this paper for the details omitted in this extended abstract.

## 2 Preliminaries

For a positive integer  $n$ , we use the standard notation  $[n] = \{1, 2, \dots, n\}$ . All logarithms in this paper are to base 2. A CNF formula  $F$  is a conjunction of clauses, denoted  $F = \bigwedge_j C_j$ , where each clause  $C$  is a disjunction of literals, denoted  $C = \bigvee_i a_i$ . Each literal  $a$  is either a propositional variable  $x$  (a *positive literal*) or its negation  $\bar{x}$  (a *negative literal*). We think of formulas and clauses as sets, so that there is no repetition and order does not matter. We consider polynomials on the same propositional variables, with the convention that, as an algebraic variable,  $x$  evaluates to 1 when it is true and to 0 when it is false. All polynomials in this paper are evaluated on 0/1-assignments, and live in the ring of real multilinear polynomials, which is the ring of real polynomials modulo the ideal generated by polynomials  $x_i^2 - x_i$  for all variables  $x_i$ . In other words, all variables in all monomials have degree at most one, and monomial multiplication is defined by  $(\prod_{i \in A} x_i) \cdot (\prod_{i \in B} x_i) = \prod_{i \in A \cup B} x_i$ .

Since sums-of-squares derivations operate with polynomial equations and inequalities, in order to reason about CNF formulas we need to encode them in this language. For a clause  $C = C^+ \vee C^-$ , where we write  $C^+$  and  $C^-$  to denote the subsets of positive and negative

literals, respectively, we define

$$S(C) = \sum_{x \in C^+} x + \sum_{\bar{x} \in C^-} (1 - x) \tag{2.1}$$

and encode  $C$  as the inequality

$$S(C) \geq 1 . \tag{2.2}$$

Clearly, a clause  $C$  is satisfied by a 0/1-assignment if and only if the same assignment satisfies the inequality  $S(C) \geq 1$ . For a variable  $x$  and a bit  $\beta \in \{0, 1\}$ , we define

$$\delta_{x=\beta} = \begin{cases} 1 - x & \text{if } \beta = 0, \\ x & \text{if } \beta = 1; \end{cases} \tag{2.3}$$

and for a sequence of variables  $\vec{x} = (x_{i_1}, \dots, x_{i_w})$  and a binary string  $\beta = (\beta_1, \dots, \beta_w)$ , we define the *indicator polynomial*

$$\delta_{\vec{x}=\beta} = \prod_{j=1}^w \delta_{x_{i_j}=\beta_j} \tag{2.4}$$

expanded out as a linear combination of monomials. That is,  $\delta_{\vec{x}=\beta}$  is the polynomial that evaluates to 1 for 0/1-assignments satisfying the equalities  $x_{i_j} = \beta_j$  for  $j = 1, \dots, w$  and to 0 for all other 0/1-assignments. We have the following useful fact.

► **Fact 2.1.** *For every sequence of variables  $\vec{x}$  the syntactic equality  $(\sum_{\beta \in \{0,1\}^w} \delta_{\vec{x}=\beta}) = 1$  holds (after cancellation of terms).*

Let  $F$  be a CNF formula over some set of variables denoted as  $Vars(F)$ , and let  $\rho$  be a *partial assignment* on  $Vars(F)$ . We write  $F \upharpoonright_\rho$  to denote the formula  $F$  *restricted* by  $\rho$ , where all clauses  $C \in F$  satisfied by  $\rho$  are removed and all literals falsified by  $\rho$  in other clauses are removed. For a polynomial  $p$  over variables  $Vars(F)$  (written, as always, as a linear combination of distinct monomials), we let  $p \upharpoonright_\rho$  denote the polynomial obtained by substituting values for assigned variables and removing monomials that evaluate to 0. We extend this definition to sets of formulas or polynomials in the obvious way by taking unions.

► **Definition 2.2** (Sums-of-squares proof system). *A sums-of-squares derivation, or SOS derivation for short, of the polynomial inequality  $p \geq 0$  from the system of polynomial constraints*

$$f_1 = 0, \dots, f_s = 0, h_1 \geq 0, \dots, h_t \geq 0 \tag{2.5}$$

is a sum

$$p = \sum_{j=1}^s g_j f_j + \sum_{j=1}^t u_j h_j + u_0 , \tag{2.6}$$

where  $g_1, \dots, g_s$  are arbitrary polynomials and each  $u_j$  is expressible as a sums of squares  $\sum_{\ell} q_{j,\ell}^2$ . A derivation of the equation  $p = 0$  is a pair of derivations of  $p \geq 0$  and  $-p \geq 0$ . A *sums-of-squares refutation* of (2.5) is a derivation of the inequality  $-1 \geq 0$  from (2.5).

The *degree* of an SOS derivation is the maximum degree among all the polynomials  $g_j f_j$ ,  $u_j h_j$ , and  $u_0$  in (2.6). The *size* of an SOS derivation is the total number of monomials (counted with repetition) in all polynomials  $g_j f_j$ ,  $u_j h_j$ , and  $u_0$  (all expanded out as linear combinations of distinct monomials). The size and degree of refuting an unsatisfiable system of polynomial constraints are defined by taking the minimum over all SOS refutations of the system with respect to the corresponding measure.

We remark that our choice of the multilinear setting is without any loss of generality and only serves to simplify the technical arguments slightly.

Let us state some useful basic properties of multilinear polynomials for later reference.

► **Proposition 2.3** (Unique multilinear representation). *Every function  $f : \{0, 1\}^n \rightarrow \mathbb{R}$  has a unique representation as a multilinear polynomial. In particular, if  $p$  is a multilinear polynomial such that  $p(\alpha) \in \{0, 1\}$  for all  $\alpha \in \{0, 1\}^n$ , then for every positive integer  $\ell$  the equality  $p^\ell = p$  holds (where this is a syntactic equality of multilinear polynomials expanded out as linear combinations of distinct monomials).*

The upper bounds in this paper are shown in the weaker proof system *resolution*, which is defined as follows. A *resolution derivation* of a clause  $D$  from a CNF formula  $F$  is a sequence of clauses  $(D_1, D_2, \dots, D_\tau)$  such that  $D_\tau = D$  and for every clause  $D_i$  it holds that it is either a clause of  $F$  (an *axiom*), or is obtained by *weakening* from some  $D_j \subseteq D_i$  for  $j < i$ , or can be inferred from two clauses  $D_\ell, D_j$ ,  $\ell < j < i$ , by the *resolution rule* that allows to derive the clause  $A \vee B$  from two clauses  $A \vee x$  and  $B \vee \bar{x}$  (where we say that  $A \vee x$  and  $B \vee \bar{x}$  are *resolved on  $x$*  to yield the *resolvent*  $A \vee B$ ). If in a resolution derivation  $(D_1, D_2, \dots, D_\tau)$  each clause  $D_j$  is only used once in a weakening or resolution step to derive some  $D_i$  for  $i > j$ , we say that the derivation is *tree-like* (such derivations may contain multiple copies of the same clause). A *resolution refutation* of  $F$ , or *resolution proof* for  $F$ , is a derivation of the empty clause (the clause containing no literals) from  $F$ .

The *width* of a clause is the number of literals in it, and the width of a CNF formula or resolution derivation is the maximal width of any clause in the formula or derivation. The *size* of a resolution derivation is the total number of clauses in it (counted with repetitions). The size and width of refuting an unsatisfiable CNF formula  $F$  is defined by taking the minimum over all resolution refutations of  $F$  with respect to the corresponding measure.

The following standard fact is easy to establish by forward induction over resolution derivations. We omit the proof.

► **Fact 2.4.** *Consider a partial assignment  $\rho$  which assigns  $\ell$  variables. Let  $A$  be the unique clause of width  $\ell$  such that  $A$  evaluates to false under  $\rho$ . If resolution can derive  $C$  in width  $w$  and size  $S$  from  $F \setminus \rho$ , then resolution can derive  $A \vee C$  in width at most  $w + \ell$  and size at most  $S + 1$  from  $F$ .*

Let us also state for the record the formal claim that SOS is more powerful than resolution in term of degree (and for constant degree also in terms of size). The next lemma is essentially Lemma 4.6 in [1], except that there the lemma is stated for the Sherali-Adams proof system. Since SOS simulates Sherali-Adams efficiently with respect to both size and degree, however, the same bounds apply also for SOS.

► **Lemma 2.5** (SOS simulation of resolution). *If a CNF formula  $F = \bigwedge_{j=1}^t C_j$  has a resolution refutation of size  $S$  and width  $w$ , then the constraints  $\{S(C_j) \geq 1\}_{j=1}^t$  as defined in (2.1) and (2.2) have an SOS refutation of size  $O(w2^w S)$  and degree at most  $w + 1$ .*

The next lemma will be useful as a subroutine when we prove upper bounds in resolution. We again omit the proof.

► **Lemma 2.6.** *Let  $k$  and  $m_1, m_2, \dots, m_k$  be positive numbers. Then the CNF formula*

consisting of the clauses

$$y_{i,0} \quad i \in [k], \quad (2.7a)$$

$$\bar{y}_{i,j-1} \vee x_{i,j} \vee y_{i,j} \quad i \in [k], j \in [m_i], \quad (2.7b)$$

$$\bar{y}_{i,m_i} \quad i \in [k], \quad (2.7c)$$

$$\bar{x}_{1,j_1} \vee \bar{x}_{2,j_2} \cdots \vee \bar{x}_{k,j_k} \quad (j_1, \dots, j_k) \in [m_1] \times \cdots \times [m_k], \quad (2.7d)$$

has a resolution refutation of width  $k + 1$  and size  $O(\prod_{i=1}^k m_i)$ .

When we construct formulas to be relativized as described in Section 1.2, it is convenient to use variables  $x_{i,\vec{j}}$ , where  $i$  ranges over some specific domain  $D$  and  $\vec{j}$  is a collection of other indices. We say that the variable  $x_{i,\vec{j}}$  mentions the element  $i \in D$ . The *domain-width* of a clause is the number of distinct elements of  $D$  mentioned by its variables. The domain-width of a CNF formula or resolution proof is defined by taking the maximum domain-width over all its clauses, and the domain-width of refuting a CNF formula  $F$  is the minimal domain-width of any resolution refutation of  $F$ . Similarly, the *domain-degree* of a monomial is the number of distinct elements in  $D$  mentioned by its variables, the domain-degree of a polynomial or SOS proof is the maximal domain-degree of any monomial in it, and the domain-degree of refuting an unsatisfiable system of polynomial constraints is defined by taking the minimum over all refutations.

### 3 A Degree Lower Bound for Clique Formulas

In this section we state and prove the formal version of Theorem 1.2, namely a lower bound for the domain-degree needed in SOS to prove that a graph  $G$  has no  $k$ -clique. Let us start by describing how we encode the  $k$ -clique problem as a CNF formula.

► **Definition 3.1** ( $k$ -clique formula). Let  $k$  be a positive integer,  $G = (V, E)$  be an undirected graph on  $N$  vertices, and  $(v_1, v_2, \dots, v_N)$  be an enumeration of  $V(G) = V$ . Then the formula  $k$ -Clique( $G$ ) consists of the clauses

$$\bar{x}_{i,u} \vee \bar{x}_{i',v} \quad i, i' \in [k], i \neq i', \{u, v\} \notin E(G), \quad (3.1a)$$

$$\bar{x}_{i,u} \vee \bar{x}_{i,v}, \quad i \in [k], u, v \in V(G), u \neq v, \quad (3.1b)$$

$$z_{i,0} \quad i \in [k], \quad (3.1c)$$

$$\bar{z}_{i,(j-1)} \vee x_{i,v_j} \vee z_{i,j} \quad i \in [k], j \in [N], \quad (3.1d)$$

$$\bar{z}_{i,N} \quad i \in [k]. \quad (3.1e)$$

The formula  $k$ -Clique( $G$ ) encodes the claim that  $G$  has a clique of size  $k$ . The intended meaning of the variable  $x_{i,v}$  for  $v \in V(G)$  is that  $v$  is the  $i$ th vertex of the clique. The variables of  $k$ -Clique( $G$ ) are indexed by  $i$  over the domain  $[k]$  and the domain-width of the formula is 2. The next proposition shows that the naive brute-force approach to decide  $k$ -Clique( $G$ ) can be carried on in resolution (and hence by Lemma 2.5 also in SOS).

► **Proposition 3.2.** *If  $G$  has no clique of size  $k$ , then  $k$ -Clique( $G$ ) has a resolution refutation of size  $O(|V|^k)$  and width  $k + 1$ .*

**Proof.** We first use the weakening rule to derive all clauses of the form

$$\bar{x}_{1,u_1} \vee \bar{x}_{2,u_2} \vee \cdots \vee \bar{x}_{k,u_k} \quad (3.2)$$

for every sequence of vertices  $(u_1, u_2, \dots, u_k)$ . This is possible since either the sequence contains a repetition or it includes two vertices with no edge between them, and in both cases this means that the clause (3.2) is a superclause of some clause of the form (3.1a). Then we derive the empty clause by applying Lemma 2.6 to the clauses (3.1c)–(3.1e) and (3.2). ◀

In order to obtain suitably hard instances of  $k$ -Clique( $G$ ) we construct a reduction from 3-XORs to  $k$ -partite graphs. It is convenient for us to describe the special case of  $k$ -clique on  $k$ -partite graphs directly as an encoding as polynomial equations and inequalities as follows next.

► **Definition 3.3** (Polynomial encoding of  $k$ -clique on  $k$ -partite graphs). For a  $k$ -partite graph  $G$  with  $V(G) = V_1 \dot{\cup} V_2 \dot{\cup} \dots \dot{\cup} V_k$  we let  $k$ -Block( $G$ ) denotes the following collection of polynomial constraints:

$$\sum_{v \in V_i} x_v = 1 \quad i \in [k], \quad (3.3a)$$

$$x_u + x_v \leq 1 \quad u \in V_i, v \in V_{i'}, i \neq i', \{u, v\} \notin E(G). \quad (3.3b)$$

► **Proposition 3.4.** Consider a  $k$ -partite graph  $G$ , where  $V(G) = V_1 \dot{\cup} V_2 \dot{\cup} \dots \dot{\cup} V_k$ . If  $k$ -Clique( $G$ ) has an SOS refutation in domain-degree  $d$ , then  $k$ -Block( $G$ ) has an SOS refutation in domain-degree  $d$ .

**Proof.** The proof is by transforming a refutation of  $k$ -Clique( $G$ ) into a refutation of  $k$ -Block( $G$ ) of the same domain-degree. To give an overview, we start with a refutation of  $k$ -Clique( $G$ ) of domain-degree  $d$  and replace its variables with polynomials of degree at most 1 mentioning only variables from  $k$ -Block( $G$ ). In this way we get an SOS refutation of domain-degree at most  $d$  from the substituted axioms of  $k$ -Clique( $G$ ). The latter polynomials are not necessarily axioms of  $k$ -Block( $G$ ), but we show that they have SOS derivations of domain-degree 1 from the axioms of  $k$ -Block( $G$ ). This concludes the proof.

The variable substitution has two steps: first we substitute every variable  $z_{i,j}$  with the linear form  $\sum_{t=j+1}^N x_{i,v_t}$ , where  $\{v_j\}_{j=1}^N$  is the enumeration of  $V(G)$  in Definition 3.1, and then we set  $x_{i,v_j}$  to 0 whenever  $v_j \notin V_i$ .

As mentioned above, we now need to give SOS derivations of domain-degree 1 of all transformed axioms in  $k$ -Clique( $G$ ) from  $k$ -Block( $G$ ). For the axioms (3.1c)–(3.1e), the SOS encoding is

$$z_{i,0} \geq 1 \quad i \in [k], \quad (3.4a)$$

$$(1 - z_{i,(j-1)}) + x_{i,v_j} + z_{i,j} \geq 1 \quad i \in [k], j \in [N], \quad (3.4b)$$

$$(1 - z_{i,N}) \geq 1 \quad i \in [k]. \quad (3.4c)$$

After the first step of the substitution the inequalities (3.4a), (3.4b) and (3.4c) become, respectively, the inequality  $\sum_{j=1}^N x_{i,v_j} \geq 1$ , and two occurrences of tautology  $1 \geq 1$ . Furthermore, after the second step of the substitution the inequality (3.4a) becomes  $\sum_{v \in V_i} x_{i,v} \geq 1$ , which is subsumed by Equation (3.3a). Each of the axioms (3.1a) and (3.1b) is encoded as

$$1 - x_{i,u} - x_{i',v} \geq 0 \quad (3.5)$$

for some pair of indices  $i, i'$  and vertices  $u, v$ . We assume that  $u \in V_i$  and  $v \in V_{i'}$ , because otherwise the variable substitution turns the inequality into either a tautology or into  $1 - x_{i,u} \geq 0$ , where the latter follows from  $(1 - x_{i,u})^2 \geq 0$  by multilinearity. If  $i \neq i'$  then



the inequality (3.5) is an axiom of  $k$ -Block( $G$ ). If that is not the case, then we can obtain  $1 - x_{i,u} - x_{i,v}$  in domain-degree 1 using the derivation

$$1 - \underbrace{\sum_{v \in V_i} x_{i,w}}_{\text{from Equation (3.3a)}} + \underbrace{\sum_{w \notin \{u,v\}} (x_{i,w})^2}_{\text{sum of squares}} = 1 - \sum_{v \in V_i} x_{i,w} + \sum_{w \notin \{u,v\}} x_{i,w} = 1 - x_{i,u} - x_{i,v} \quad (3.6)$$

where the first identity holds by multilinearity. The proposition follows. ◀

What we want to do now is to prove a domain-degree lower bound for instances of  $k$ -Block( $G$ ) where the graph  $G$  is obtained by a reduction from (unsatisfiable) sets of  $\mathbb{F}_2$ -linear equations. We rely on the version of Grigoriev’s degree lower bound [13] shown by Schoenebeck [29], which is conveniently stated for random 3-XOR formulas as encoded next.

► **Definition 3.5** (Polynomial encoding of random 3-XOR). A random 3-XOR formula  $\phi$  represents a system of  $\Delta n$  linear equations modulo 2 defined over  $n$  variables. Each equation is sampled at random among all equations of the form  $x \oplus y \oplus z = b$  as follows:  $x, y, z$  are sampled uniformly without replacement from the set of  $n$  variables and  $b$  is sampled uniformly in  $\{0, 1\}$ . The polynomial encoding of any such linear equation modulo 2 is

$$(1 - x)(1 - y)z = 0 \quad (3.7a)$$

$$(1 - x)y(1 - z) = 0 \quad (3.7b)$$

$$x(1 - y)(1 - z) = 0 \quad (3.7c)$$

$$xyz = 0 \quad (3.7d)$$

when  $b = 0$  and

$$(1 - x)(1 - y)(1 - z) = 0 \quad (3.7e)$$

$$xy(1 - z) = 0 \quad (3.7f)$$

$$x(1 - y)z = 0 \quad (3.7g)$$

$$(1 - x)yz = 0 \quad (3.7h)$$

when  $b = 1$ .

Fixing  $\delta = 1/4$  and  $\Delta = 8$  in [29] we have the following theorem.

► **Theorem 3.6** ([29]). *There exists an  $\alpha, 0 < \alpha < 1$ , such that for every  $\epsilon > 0$  there exists an  $n_\epsilon \in \mathbb{N}$  such that a random 3-XOR formula  $\phi$  in  $n \geq n_\epsilon$  variables and  $8n$  constraints has the following properties with probability at least  $1 - \epsilon$ .*

1. *At most  $6n$  parity constraints of  $\phi$  can be simultaneously satisfied.*
2. *Any sums-of-squares refutation of  $\phi$  requires degree  $\alpha n$ .*

Now we are ready to describe how to transform a 3-XOR formula  $\phi$  into a  $k$ -partite graph  $G_\phi^k$  that has a clique of size  $k$  if and only if  $\phi$  is satisfiable.

► **Definition 3.7** (3-XOR graph). Given  $k \in \mathbb{N}$  and a 3-XOR formula  $\phi$  with  $8n$  constraints over  $n$  variables, where we assume for simplicity that  $k$  divides  $8n$ , we construct a 3-XOR graph  $G_\phi^k$  as follows.

We arbitrarily split the formula  $\phi$  into  $k$  linear systems with  $8n/k$  constraints each, denoted as  $\phi_1, \phi_2, \dots, \phi_k$ . For each  $\phi_i$  we let  $V_i$  be a set of at most  $N \leq 2^{24n/k}$  vertices labelled by all possible assignments to the at most  $24n/k$  variables appearing in  $\phi_i$ . For

two distinct vertices  $u \in V_i$  and  $v \in V_{i'}$ , there is an edge between  $u$  and  $v$  in  $G_\phi^k$  if the two assignments corresponding to  $u$  and  $v$  are compatible, i.e., when they assign the same values to the common variables, and also the union of the two assignments does not violate any constraint in  $\phi$ . (In particular, each  $V_i$  is an independent set, since two distinct assignments to the same set of variables are not compatible.)

The key property of the reduction in Definition 3.7 is that it allows small domain-degree refutations of  $k\text{-Block}(G_\phi^k)$  to be converted into small degree refutations of  $\phi$ .

► **Lemma 3.8.** *If  $k\text{-Block}(G_\phi^k)$  has an SOS refutation of domain-degree  $d$ , then  $\phi$  has an SOS refutation of degree  $24dn/k$ .*

**Proof.** Again we start by giving an overview of the proof, which works by transforming a refutation of  $k\text{-Block}(G_\phi^k)$  of domain-degree  $d$  into a refutation of  $\phi$  of degree  $24dn/k$ .

Given a refutation of  $k\text{-Block}(G_\phi^k)$  of domain-degree  $d$ , we replace every variable  $x_v$  with a polynomial over the variables of  $\phi$ . In this way we get an SOS refutation from the polynomials corresponding to the substituted axioms of  $k\text{-Block}(G_\phi^k)$ . The latter polynomials need not be axioms of  $\phi$ , but we show that they can be efficiently derived in SOS from  $\phi$ . We thus obtain an SOS refutation of  $\phi$ , the degree of which is easily verified to be as in the statement of the lemma.

We now describe the substitution in detail. Consider a block  $V_i$  and suppose that the corresponding 3-XOR formula  $\phi_i$  mentions  $t$  variables. Let us write  $\vec{x}$  to denote this set of variables. Then every vertex  $v \in V_i$  represents an assignment  $\beta \in \{0, 1\}^t$  to  $\vec{x}$ . In what follows, we denote the indicator polynomial  $\delta_{\vec{x}=\beta}$  in (2.4) by  $\delta_v$  for brevity, and we substitute for each variable  $x_v$  the polynomial  $\delta_v$  of degree  $t \leq 24n/k$ .

Before the substitution each monomial in the original refutation has domain-degree at most  $d$  by assumption. Two important observations are that  $(\delta_v)^2 = \delta_v$  for every  $v \in V_i$  and that  $\delta_u \delta_v = 0$  for every two distinct  $u, v$  in the same block  $V_i$ . Therefore, after the substitution each monomial is either identically zero or the product of at most  $d$  indicator polynomials, and hence its degree is at most  $24dn/k$ .

In order to complete the proof outline above, we now need to present SOS derivations starting from the 3-XOR constraints of  $\phi$  of all polynomial constraints resulting from the substitutions in the axioms of  $k\text{-Block}(G_\phi^k)$  described above, and to do so in degree at most  $24n/k$ .

Let us first look at the axioms (3.3a). By Fact 2.1, the identity

$$\sum_{v \in V_i} \delta_v = \sum_{\beta \in \{0,1\}^t} \delta_{\vec{x}=\beta} = 1 \quad (3.8)$$

holds syntactically, so substitutions in axioms of the form (3.3a) result in tautologies  $1 = 1$ .

The remaining axioms of  $k\text{-Block}(G_\phi^k)$  in (3.3b) have the form  $x_u + x_v \leq 1$  for non-edges  $(u, v)$  between vertices in different blocks. By construction of  $G_\phi^k$  the reason  $u$  and  $v$  are not connected is either that the partial assignments corresponding to the two vertices are incompatible, or that their union violates some constraint in  $\phi$ .

In the first case,  $1 - \delta_u - \delta_v \geq 0$  is an SOS axiom because of the identity

$$(1 - \delta_u - \delta_v)^2 = 1 - \delta_u - \delta_v, \quad (3.9)$$

which follows from the observation that  $\delta_u$  and  $\delta_v$  are the indicator polynomials of two incompatible assignments and cannot evaluate to 1 simultaneously, and so  $(1 - \delta_u - \delta_v)$  evaluates to either 0 or 1 and is identical to its square by Proposition 2.3. The degree of (3.9) is  $24n/k$ .

In the second case, the two assignments corresponding to  $u$  and  $v$  are compatible but their union violates some initial equation  $f = 0$  of the form (3.7a)–(3.7h). Any such  $f$  is a degree-3 indicator polynomial which evaluates to 1 whenever the assignment satisfies the equations  $\delta_u \delta_v = 1$ . This means that  $\delta_u \delta_v$  contains  $f$  as a factor. We factorize  $f$  as  $f_u f_v$  so that  $\delta_u = f_u \delta'_u$  and  $\delta_v = f_v \delta'_v$ . Given this notation, we can derive  $0 \leq 1 - \delta_u - \delta_v$  using the identity

$$(1 - f_u - f_v)^2 + (f_u - \delta_u)^2 + (f_v - \delta_v)^2 - 2f_u f_v = 1 - \delta_u - \delta_v \tag{3.10}$$

of degree at most  $24n/k$ . The lemma follows. ◀

Now we can put together all the material in this section to prove a formal version of Theorem 1.2 as stated next.

► **Theorem 3.9.** *There are universal constants  $N_0 \in \mathbb{N}^+$  and  $\alpha_0, 0 < \alpha_0 < 1$ , such that for every  $k \geq 1$  there exists a graph  $G_k$  with at most  $kN_0 = O(k)$  vertices and a 3-CNF formula  $k$ -Clique( $G_k$ ) of size polynomial in  $k$  with the following properties:*

1. Resolution can refute  $k$ -Clique( $G_k$ ) in size  $2^{O(k \log k)}$  and width  $k + 1$ .
2. Any SOS refutation of  $k$ -Clique( $G_k$ ) requires domain-degree  $\alpha_0 k$ .

**Proof.** Fix any positive  $\epsilon < 1$  and let  $N_0 = 2^{24n_\epsilon}$ ,  $\alpha_0 = \frac{\alpha}{24}$  and  $n = kn_\epsilon$ , where  $n_\epsilon$  and  $\alpha$  are the universal constants from Theorem 3.6. To build the graph  $G_k$  we take a 3-XOR formula  $\phi$  on  $n$  variables and  $8n$  equations from the distribution in Definition 3.5. Since  $n \geq n_\epsilon$ , Theorem 3.6 implies that there is a formula in the support of the distribution that is unsatisfiable and that requires degree  $\alpha n$  to be refuted in SOS. We fix  $\phi$  to be that formula and let  $G_k$  be the graph  $G_\phi^k$  constructed as in Definition 3.7. Then  $G_\phi^k$  is  $k$ -partite, with each part having at most  $2^{24n/k} = N_0$  vertices, and the graph has no  $k$ -clique because otherwise  $\phi$  would be satisfiable.

Suppose that there is an SOS refutation of  $k$ -Clique( $G_\phi^k$ ) of domain-degree  $d$ . We want to argue that  $d \geq \alpha_0 k$ . Since  $G_\phi^k$  is  $k$ -partite, by Proposition 3.4 the formula  $k$ -Block( $G_\phi^k$ ) also has an SOS refutation in domain-degree  $d$ . By Lemma 3.8, this in turn yields an SOS refutation of  $\phi$  in degree  $24dn/k$ . Now Theorem 3.6 implies that  $24dn/k \geq \alpha n$ , and hence  $d \geq \frac{\alpha}{24} k = \alpha_0 k$ .

To conclude the proof, we can just observe that the resolution width and size upper bounds are a direct application of Proposition 3.2. ◀

## 4 Size Lower Bounds from Relativization

Using the material developed in Section 3, we can now describe how to *relativize* formulas in order to amplify degree lower bounds to size lower bounds in SOS. This method works for formulas that are “symmetric” in a certain sense, and so we start by explaining exactly what is meant by this.

► **Definition 4.1** (Symmetric formula). Consider a CNF formula  $F$  on variables  $x_{i,\vec{j}}$ , where  $i$  is an index in some domain  $D$  and  $\vec{j}$  denotes a collection of other indices. For every subset of indices  $\vec{i} = \{i_1, i_2, \dots, i_s\} \subseteq D$  we identify the subformula  $F_{\vec{i}}$  of  $F$  such that each clause  $C \in F_{\vec{i}}$  mentions *exactly* the indices in  $\vec{i}$ , so that a formula  $F$  of domain-width  $d$  can be written as

$$F = \bigwedge_{s=0}^d \bigwedge_{\substack{\vec{i} \subseteq D \\ |\vec{i}|=s}} F_{\vec{i}} \tag{4.1}$$

We say that  $F$  is *symmetric with respect to  $D$*  if it is invariant with respect to permutations of  $D$ , i.e., if for every  $F_{\vec{v}} \subseteq F$  it also holds that  $F_{\pi(\vec{v})} \subseteq F$ , where  $\pi$  is any permutation on  $D$  and  $\pi(\vec{v})$  is the set of images of the indices in  $\vec{v}$ . Phrased differently,  $F$  is symmetric with respect to  $D$  if for any permutation  $\pi$  on  $D$  the *syntactic* equality  $F = \bigwedge_{\vec{v} \subseteq D} F_{\pi(\vec{v})}$  holds (where we recall that we treat CNF formulas as sets of clauses). We apply this terminology for systems of polynomial equations and inequalities in the same way.

► **Observation 4.2.** *The  $k$ -Clique( $G$ ) formula in Definition 3.1 over variables  $x_{i,v}$  is symmetric with respect to the indices  $i \in [k]$ .*

Starting with any formula  $F$  symmetric with respect to a domain  $D$ , we can build a family of similar formulas by varying the size of the domain. If  $F$  has domain-width  $d$ , then for each  $s$ ,  $0 \leq s \leq d$ , the subformulas  $F_{\vec{v}}$  with  $|\vec{v}| = s$  in (4.1) are the same up to renaming of the domain indices in  $\vec{v}$ . Hence, we can arbitrarily pick one such subformula to represent them all, and denote it as  $F_s$ . The formulas  $\{F_s\}_{s=0}^d$  are completely determined by  $F$ , and together with  $D$  they in turn completely determine  $F$ . Using this observation, we can generalize the formula  $F$  over domain  $D$  to any domain  $D'$  with  $|D'| \geq d$  by defining  $F[D']$  to be the formula

$$F[D'] = \bigwedge_{s=0}^d \bigwedge_{\substack{\vec{v} \subseteq D \\ |\vec{v}|=s}} F_{\vec{v}} \quad , \quad (4.2)$$

where each  $F_{\vec{v}}$  for  $|\vec{v}| = s$  is an isomorphic copy of  $F_s$  with its domain indices renamed according to  $\vec{v}$ . Let us state some simple but useful facts that can be read off directly from (4.2):

1. For any formula  $F$  of domain-width  $d$  symmetric with respect to domain  $D$ , it holds that  $F[D]$  is (syntactically) equal to  $F$ .
2. For any domains  $D', D''$  with  $|D'| = |D''| \geq d$ , the two formulas  $F[D']$  and  $F[D'']$  are isomorphic.
3. For any  $D'' \supseteq D'$  with  $|D'| \geq d$ , the formula  $F[D'']$  contains many isomorphic copies of  $F[D']$ .

When we want to emphasize the domain  $D$  of a formula  $F$  in what follows, we will denote the formula  $F$  as  $F[D]$ . When the domain is  $D = [t]$ , we abuse notation slightly and write  $F[t]$  instead of  $F[[t]]$ . As discussed above, from a symmetric formula  $F$  of domain-width  $d$  we can obtain a well-defined sequence of formulas  $F[t]$  for all  $t \geq d$ . We say that the *unsatisfiability threshold* of such a sequence of formulas is the least  $t$  such that  $F[t]$  is unsatisfiable.

## 4.1 Relativization of Symmetric Formulas

Given a formula  $F = F[m]$  symmetric with respect to  $[m]$  and a parameter  $k < m$ , we now want to define the  *$k$ -relativization* of  $F[m]$ , which is intended to encode the claim that there exists a subset  $D \subseteq [m]$  of size  $|D| \geq k$  such that the subformula  $F[D] \subseteq F[m]$  is satisfiable. We remark that a CNF formula encoding such a claim will be unsatisfiable when  $k$  is at least the unsatisfiability threshold of  $F$ .

In order to express the existence of the subset  $D$  we use *selectors*  $s_1, s_2, \dots, s_m$  as indicators of membership in the subset and encode the constraint on the subset size  $|D| = \sum_{i=1}^m s_i \geq k$  as described in the next definition.

► **Definition 4.3.** The *threshold- $k$  formula* for variables  $\vec{s} = \{s_1, \dots, s_m\}$  is the 3-CNF formula  $\text{Thr}_k(\vec{s})$  that consists of the clauses

$$y_{\ell,0} \quad \ell \in [k], \quad (4.3a)$$

$$\bar{y}_{\ell,i-1} \vee p_{\ell,i} \vee y_{\ell,i} \quad \ell \in [k], i \in [m], \quad (4.3b)$$

$$\bar{y}_{\ell,m} \quad i \in [m], \quad (4.3c)$$

$$\bar{p}_{\ell,i} \vee \bar{p}_{\ell',i} \quad \ell, \ell' \in [k], \ell \neq \ell', i \in [m], \quad (4.3d)$$

$$\bar{p}_{\ell,i} \vee s_i \quad \ell \in [k], i \in [m]. \quad (4.3e)$$

To see that  $\text{Thr}_k(\vec{s})$  indeed enforces a cardinality constraint, note that the variables  $p_{\ell,i}$  encode a mapping between  $[k]$  and  $[m]$  (with  $p_{\ell,i}$  being true if and only if  $\ell$  maps to  $i$ ). We will need the following properties of the threshold formula.

► **Observation 4.4.** *The formula  $\text{Thr}_k(\vec{s})$  in Definition 4.3 has the following properties:*

1.  $\text{Thr}_k(\vec{s})$  has size polynomial in both  $k$  and  $m$ .
2. For any partial assignment to  $\vec{s}$  with at least  $k$  ones there is an assignment to the extension variables that satisfies  $\text{Thr}_k(\vec{s})$ .
3. There is a resolution refutation of the set of clauses  $\text{Thr}_k(\vec{s}) \cup \{\bigvee_{i \in D} \bar{s}_i \mid D \subseteq [m], |D| = k\}$  of size  $O(km^k)$  and width  $k + 1$ .

**Proof.** The first two items are immediate. In order to show the third item we can first derive each clause  $\bar{p}_{1,i_1} \vee \dots \vee \bar{p}_{k,i_k}$  by resolving  $\bar{s}_{i_1} \vee \dots \vee \bar{s}_{i_k}$  with clauses of the form (4.3e), and then apply Lemma 2.6. ◀

Using the formula in Definition 4.3 to encode cardinality constraints on subsets, we can now define formally what we mean by the relativization of a symmetric formula.

► **Definition 4.5 (Relativization).** Given a CNF formula  $F$  symmetric with respect to a domain  $[m]$  and a parameter  $k < m$ , the  *$k$ -relativization* (or  *$k$ -relativized formula*)  $F[k; m]$  is the formula consisting of

1. the threshold formula  $\text{Thr}_k(\vec{s})$  over selectors  $\vec{s} = \{s_1, \dots, s_m\}$ ;
2. a *selectable clause*  $\bar{s}_{i_1} \vee \dots \vee \bar{s}_{i_s} \vee C$  for each clause  $C \in F[m]$ , where  $\{i_1, i_2, \dots, i_s\}$  are the indices mentioned by  $C$ .

Since we are dealing with refutations of unsatisfiable formulas, it will always be the case that the parameter  $k$  in Definition 4.5 is at least the unsatisfiability threshold of  $F$ . An important property of relativized formulas is that the hardness of  $F[k; m]$  scales nicely with  $m$ . In particular, if  $F[k]$  is not too hard, then the relativization  $F[k; m]$  also is not too hard.

► **Proposition 4.6.** *If  $F[k]$  has a resolution refutation of size  $S$  and width  $w$ , then  $F[k; m]$  has a resolution refutation of size  $S \cdot \binom{m}{k} + O(km^k)$  and width  $w + k$ .*

**Proof.** For every set  $D \subseteq [m]$  with  $|D| = k$  we show how to derive

$$\bigvee_{i \in D} \bar{s}_i \quad (4.4)$$

in size  $S + 1$  and width  $w + k$  from  $F[k; m]$ . Without loss of generality (because of symmetry) we assume that  $D = [k]$ , so that we want to derive  $\bar{s}_1 \vee \dots \vee \bar{s}_k$ . Consider the assignment  $\rho = \{s_1 = 1, \dots, s_k = 1\}$ . In the restricted formula  $F[k; m] \upharpoonright_\rho$  the selectable clauses in Definition 4.5, item 2, with all indices in  $[k]$  become the clauses of  $F[k]$ , which has a

refutation of size  $S$  and width  $w$ . Thus the clause  $\bar{s}_1 \vee \cdots \vee \bar{s}_k$  can be derived in size  $S + 1$  and width  $w + k$  from  $F[k; m]$  by Fact 2.4. After we have derived all clauses of the form (4.4) in this way, we can obtain the empty clause in width  $k + 1$  and in size at most  $O(km^k)$  using Observation 4.4.  $\blacktriangleleft$

## 4.2 Random Restrictions and Size Lower Bounds

To prove size lower bounds on refutations of relativized formulas  $F[k; m]$  we use random restrictions sampled as follows.

► **Definition 4.7** (Random restrictions for relativized formulas). Given a relativized formula  $F[k; m]$ , we define a distribution  $\mathcal{R}$  of partial assignments over the variables of this formula by the following process.

1. Pick uniformly at random a set  $D \subseteq [m]$  of size  $k$ .
2. Fix  $s_i$  to 1 if  $i \in D$  and to 0 otherwise.
3. Extend this to any assignment to the remaining variables of the formula  $\text{Thr}_k(\vec{s})$  that satisfies this threshold formula.
4. For every variable  $x_{i,\vec{j}}$  that has index  $i \notin D$ , fix  $x_{i,\vec{j}}$  to 0 or 1 uniformly and independently at random.
5. All remaining variables  $x_{i,\vec{j}}$  for the indices  $i \in D$  are left unset.

It is straightforward to verify that the distribution  $\mathcal{R}$  is constructed in such a way as to give us back  $F[k]$  from  $F[k; m]$ .

► **Observation 4.8.** *For any relativized formula  $F[k; m]$  and any  $\rho \in \mathcal{R}$  it holds that  $F[k; m] \upharpoonright_\rho$  is equal to  $F[k]$  up to renaming of variables.*

The key technical ingredient in the size lower bound on sums-of-squares proofs is the following property of the distribution  $\mathcal{R}$ , which was proven in [2, 1] but is rephrased below using the notation and terminology in this paper.

► **Lemma 4.9** ([1, 2]). *Let  $k, \ell, m$  be positive integers such that  $m \geq 16$  and  $\ell \leq k \leq m/(4 \log m)$ . Let  $M$  be a monomial over the variables of  $F[k; m]$  and let  $\rho$  be a random restriction sampled from the distribution  $\mathcal{R}$  in Definition 4.7. Then the domain-degree of  $M \upharpoonright_\rho$  is less than  $\ell$  with probability at least  $1 - (4k \log m)^k / m^\ell$ .*

Using Lemma 4.9, it is now straightforward to show that relativization amplifies degree lower bounds to size lower bounds.

► **Theorem 4.10.** *Let  $k, \ell, m$  be positive integers such that  $m \geq 16$  and  $\ell \leq k \leq m/(4 \log m)$ . If the CNF formula  $F[k]$  requires sums-of-squares refutations of domain-degree  $\ell$ , then the relativized formula  $F[k; m]$  requires sums-of-squares refutations of size  $m^\ell / (4k \log m)^k$ .*

**Proof.** Suppose that there is a sums-of-squares refutation of  $F[k; m]$  in size  $S$ , i.e., containing  $S$  monomials. For  $\rho$  sampled from  $\mathcal{R}$ , we see that the probability that some monomial in the refutation restricted by  $\rho$  has domain-degree at least  $\ell$  is at most

$$S \cdot \frac{(4k \log m)^k}{m^\ell} \tag{4.5}$$

by appealing to Lemma 4.9 and taking a union bound.

As noted in Observation 4.8, the formula  $F[k; m] \upharpoonright_\rho$  is equal to  $F[k]$  up to renaming of variables, and so it cannot have a refutation of domain-degree  $\ell$  or less. This implies that the bound on the probability (4.5) is greater than one, and thus we obtain

$$S > \frac{m^\ell}{(4k \log m)^k}, \quad (4.6)$$

which proves the theorem.  $\blacktriangleleft$

Putting everything together, we can establish the formal version of our main results in Theorem 1.1 as follows.

► **Theorem 4.11.** *Let  $k = k(m)$  be any monotone non-decreasing integer-valued function such that  $k(m) \leq m/(4 \log m)$ . Then there is a family of 4-CNF formulas  $\{F_{m,k}\}_{m \geq 1}$  with  $O(km^2)$  clauses over  $O(km)$  variables such that:*

1. *Resolution can refute  $F_{m,k}$  in size  $k^{O(k)}m^k$  and width  $2k + 1$ .*
2. *Any sums-of-squares refutation of  $F_{m,k}$  requires size  $\Omega(m^{\alpha_0 k}/(4k \log m)^k)$ , where  $\alpha_0$  is a universal constant.*

**Proof.** Let  $G$  be a graph with properties as in Theorem 3.9 and let  $F[k]$  be the CNF formula  $k$ -Clique( $G$ ) in Definition 3.1. Since  $F[k]$  is symmetric, we can relativize it as in Definition 4.5 to obtain  $F[k; m]$ , which will be our 4-CNF formula  $F_{m,k}$ . Theorem 3.9 says that  $F[k]$  has a resolution refutation of size  $k^{O(k)}$  and width  $k + 1$ , and appealing to Proposition 4.6 we get a resolution refutation of  $F_{m,k}$  in size  $k^{O(k)}m^k$  and width  $2k + 1$ . Since we have a domain-degree lower bound of  $\alpha_0 k$  for refuting  $F[k]$  according to Theorem 3.9, we can use Theorem 4.10 to deduce that the required size to refute  $F_{m,k}$  in sums-of-squares is at least  $\Omega(m^{\alpha_0 k}/(4k \log m)^k)$ . The theorem follows.  $\blacktriangleleft$

We remark that straightforward calculations show that when  $k(m) = O(m^\delta)$  for  $\delta < \alpha_0$  the upper bound in Theorem 4.11 is  $m^{O(k)}$  and the lower bound is  $m^{\Omega(k)}$ .

## 5 Concluding Remarks

In this paper, we show that using Lasserre semidefinite programming relaxations to find degree- $d$  sums-of-squares proofs is optimal up to constant factors in the exponent of the running time. More precisely, we show that there are constant-width CNF formulas on  $n$  variables that are refutable in sums-of-squares in degree  $d$  but require proofs of size  $n^{\Omega(d)}$ .

As for so many other results for the sums-of-squares proof system, in the end our proof boils down to a reduction from 3-XOR using Schoenebeck's version [29] of Grigoriev's degree lower bound [13]. It would be very interesting to obtain other SOS degree lower bounds by different means than by reducing from Grigoriev's results for 3-XOR and knapsack.

Another interesting problem would be to prove average-case SOS degree lower bound for  $k$ -clique formulas over Erdős-Rényi random graphs, or size lower bounds for (non-relativized)  $k$ -clique formulas over any graphs. In this context, it might be worth to point out that the problem of establishing proof size lower bounds for  $k$ -clique formulas for constant  $k$ , which has been discussed, for instance, in [8], still remains open even for the resolution proof system (although lower bounds have been shown for tree-like resolution in [7] and for full resolution for a version of clique formulas using a different encoding more amenable to lower bound techniques in [23]).



**Acknowledgements.** We are grateful to Albert Atserias for numerous discussions about (and explanations of) Lasserre/SOS and other LP and SDP hierarchies, as well as for help with correcting some references in a preliminary version of this manuscript. We thank Per Austrin for valuable suggestions and feedback during the initial stages of this work, and Michael Forbes for comments on an early version of the overall proof construction. Finally, we are thankful for the comments from the anonymous reviewers, which helped improve the exposition in this paper considerably.

The authors were funded by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007–2013) / ERC grant agreement no. 279611. The second author was also supported by Swedish Research Council grants 621-2010-4797 and 621-2012-5645.

---

### References

- 1 Albert Atserias, Massimo Lauria, and Jakob Nordström. Narrow proofs may be maximally long. Technical Report TR14-118, Electronic Colloquium on Computational Complexity (ECCC), September 2014. Preliminary version appeared in *CCC'14*.
- 2 Albert Atserias, Moritz Müller, and Sergi Oliva. Lower bounds for DNF-refutations of a relativized weak pigeonhole principle. In *Proceedings of the 28th Annual IEEE Conference on Computational Complexity (CCC'13)*, pages 109–120, June 2013.
- 3 Boaz Barak, Fernando G. S. L. Brandão, Aram Wettroth Harrow, Jonathan A. Kelner, David Steurer, and Yuan Zhou. Hypercontractivity, sum-of-squares proofs, and their applications. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC'12)*, pages 307–326, May 2012.
- 4 Boaz Barak and David Steurer. Sum-of-squares proofs and the quest toward optimal algorithms. Technical Report TR14-059, Electronic Colloquium on Computational Complexity (ECCC), April 2014.
- 5 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.
- 6 Paul Beame, Toniann Pitassi, and Nathan Segerlind. Lower bounds for Lovász–Schrijver systems and beyond follow from multiparty communication complexity. *SIAM Journal on Computing*, 37(3):845–869, 2007. Preliminary version appeared in *ICALP'05*.
- 7 Olaf Beyersdorff, Nicola Galesi, and Massimo Lauria. Parameterized complexity of DPLL search procedures. *ACM Transactions on Computational Logic*, 14(3):20:1–20:21, August 2013. Preliminary version appeared in *SAT'11*.
- 8 Olaf Beyersdorff, Nicola Galesi, Massimo Lauria, and Alexander A. Razborov. Parameterized bounded-depth Frege is not optimal. *ACM Transactions on Computation Theory*, 4:7:1–7:16, September 2012. Preliminary version appeared in *ICALP'11*.
- 9 Stefan Dantchev. Relativisation provides natural separations for resolution-based proof systems. In *Proceedings of the 1st International Computer Science Symposium in Russia (CSR'06)*, volume 3967 of *Lecture Notes in Computer Science*, pages 147–158. Springer, June 2006.
- 10 Stefan Dantchev and Barnaby Martin. Relativization makes contradictions harder for resolution. *Annals of Pure and Applied Logic*, 165(3):837–857, March 2014.
- 11 Stefan Dantchev and Søren Riis. On relativisation and complexity gap for resolution-based proof systems. In *Proceedings of the 17th International Workshop on Computer Science Logic (CSL'03)*, volume 2803 of *Lecture Notes in Computer Science*, pages 142–154. Springer, August 2003.

- 12 Dima Grigoriev. Complexity of Positivstellensatz proofs for the knapsack. *Computational Complexity*, 10(2):139–154, December 2001.
- 13 Dima Grigoriev. Linear lower bound on degrees of Positivstellensatz calculus proofs for the parity. *Theoretical Computer Science*, 259(1–2):613–622, May 2001.
- 14 Dima Grigoriev, Edward A. Hirsch, and Dmitrii V. Pasechnik. Exponential lower bound for static semi-algebraic proofs. In *Proceedings of the 29th International Colloquium on Automata, Languages and Programming (ICALP'02)*, volume 2380 of *Lecture Notes in Computer Science*, pages 257–268. Springer, July 2002.
- 15 Dima Grigoriev and Nicolai Vorobjov. Complexity of Null- and Positivstellensatz proofs. *Annals of Pure and Applied Logic*, 113(1-3):153–160, December 2001.
- 16 Mika Göös and Toniann Pitassi. Communication lower bounds via critical block sensitivity. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC'14)*, pages 847–856, May 2014.
- 17 Subhash Khot. On the power of unique 2-prover 1-round games. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing (STOC'02)*, pages 767–775, May 2002.
- 18 Arist Kojevnikov and Dmitry Itsykson. Lower bounds of static Lovász–Schrijver calculus proofs for Tseitin tautologies. In *Proceedings of the 33rd International Colloquium on Automata, Languages and Programming (ICALP'06)*, volume 4051 of *Lecture Notes in Computer Science*, pages 323–334. Springer, July 2006.
- 19 Jan Krajížek. Combinatorics of first order structures and propositional proof systems. *Archive for Mathematical Logic*, 43(4):427–441, May 2004.
- 20 Jean-Louis Krivine. Anneaux préordonnés. *Journal d'Analyse Mathématique*, 12(1):307–326, 1964.
- 21 Jean B. Lasserre. An explicit exact SDP relaxation for nonlinear 0-1 programs. In *Proceedings of the 8th International Conference on Integer Programming and Combinatorial Optimization (IPCO'01)*, volume 2081 of *Lecture Notes in Computer Science*, pages 293–303. Springer, June 2001.
- 22 Massimo Lauria and Jakob Nordström. Tight size-degree bounds for sums-of-squares proofs. Technical Report TR15-053, Electronic Colloquium on Computational Complexity (ECCC), April 2015.
- 23 Massimo Lauria, Pavel Pudlák, Vojtěch Rödl, and Neil Thapen. The complexity of proving that a graph is Ramsey. In *Proceedings of the 40th International Colloquium on Automata, Languages and Programming (ICALP'13)*, volume 7965 of *Lecture Notes in Computer Science*, pages 684–695. Springer, July 2013.
- 24 James R. Lee, Prasad Raghavendra, David Steurer, and Ning Tan. On the power of symmetric LP and SDP relaxations. In *Proceedings of the 29th Annual IEEE Conference on Computational Complexity (CCC'14)*, pages 13–21, June 2014.
- 25 Yurii Nesterov. Squared functional systems and optimization problems. In H. Frenk, K. Roos, T. Terlaky, and S. Zhang, editors, *High Performance Optimization*, pages 405–440. Kluwer Academic Publisher, 2000.
- 26 Ryan O'Donnell and Yuan Zhou. Approximability and proof complexity. In *Proceedings of the 24th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'13)*, pages 1537–1556, January 2013.
- 27 Pablo A. Parrilo. *Structured Semidefinite Programs and Semialgebraic Geometry Methods in Robustness and Optimization*. PhD thesis, California Institute of Technology, May 2000. Available at <http://resolver.caltech.edu/CaltechETD:etd-05062004-055516>.
- 28 Toniann Pitassi and Nathan Segerlind. Exponential lower bounds and integrality gaps for tree-like Lovász–Schrijver procedures. *SIAM Journal on Computing*, 41(1):128–159, 2012. Preliminary version appeared in *SODA'09*.

- 29 Grant Schoenebeck. Linear level Lasserre lower bounds for certain  $k$ -CSPs. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS'08)*, pages 593–602, October 2008.
- 30 N.Z. Shor. An approach to obtaining global extremums in polynomial mathematical programming problems. *Cybernetics*, 23(5):695–700, 1987. Translated from *Kibernetika*, No. 5, pages 102–106, 1987.
- 31 Gilbert Stengle. A Nullstellensatz and a Positivstellensatz in semialgebraic geometry. *Mathematische Annalen*, 207(2):87–97, 1973.
- 32 Madhur Tulsiani. CSP gaps and reductions in the Lasserre hierarchy. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing (STOC'09)*, pages 303–312, June 2009.

# A Generalized Method for Proving Polynomial Calculus Degree Lower Bounds

Mladen Mikša and Jakob Nordström

KTH Royal Institute of Technology  
SE-100 44 Stockholm, Sweden

---

## Abstract

We study the problem of obtaining lower bounds for polynomial calculus (PC) and polynomial calculus resolution (PCR) on proof degree, and hence by [Impagliazzo et al. '99] also on proof size. [Alekhovich and Razborov '03] established that if the clause-variable incidence graph of a CNF formula  $F$  is a good enough expander, then proving that  $F$  is unsatisfiable requires high PC/PCR degree. We further develop the techniques in [AR03] to show that if one can “cluster” clauses and variables in a way that “respects the structure” of the formula in a certain sense, then it is sufficient that the incidence graph of this clustered version is an expander. As a corollary of this, we prove that the functional pigeonhole principle (FPHP) formulas require high PC/PCR degree when restricted to constant-degree expander graphs. This answers an open question in [Razborov '02], and also implies that the standard CNF encoding of the FPHP formulas require exponential proof size in polynomial calculus resolution. Thus, while Onto-FPHP formulas are easy for polynomial calculus, as shown in [Riis '93], both FPHP and Onto-PHP formulas are hard even when restricted to bounded-degree expanders.

**1998 ACM Subject Classification** F.2.2 [Analysis of Algorithms and Problem Complexity] Non-numerical Algorithms and Problems – Complexity of proof procedures, F.1.3 [Computation by Abstract Devices] Complexity Measures and Classes, I.2.3 [Artificial Intelligence] Deduction and Theorem Proving, F.4.1 [Mathematical Logic and Formal Languages] Mathematical Logic – computational logic

**Keywords and phrases** Proof complexity, polynomial calculus, polynomial calculus resolution, PCR, degree, size, functional pigeonhole principle, lower bound

**Digital Object Identifier** 10.4230/LIPIcs.CCC.2015.467

## 1 Introduction

In one sentence, proof complexity studies how hard it is to certify the unsatisfiability of formulas in conjunctive normal form (CNF). In its most general form, this is the question of whether  $\text{coNP}$  can be separated from  $\text{NP}$  or not, and as such it still appears almost completely out of reach. However, if one instead focuses on concrete proof systems, which can be thought of as restricted models of (nondeterministic) computation, then fruitful study is possible.

Perhaps the most well-studied proof system in proof complexity is *resolution* [6], in which one derives new disjunctive clauses from a CNF formula until an explicit contradiction is reached, and for which numerous exponential lower bounds on proof size have been shown (starting with [8, 14, 29]). Many of these lower bounds can be established by instead studying the *width* of proofs, i.e., the size of a largest clause appearing in the proofs, and arguing that any resolution proof for a certain formula must contain a large clause. It then follows from a result by Ben-Sasson and Wigderson [5] that any resolution proof must also consist of very many clauses. Research since [5] has led to a well-developed machinery for showing width lower bounds, and hence also size lower bounds.



© Mladen Mikša and Jakob Nordström;  
licensed under Creative Commons License CC-BY  
30th Conference on Computational Complexity (CCC'15).

Editor: David Zuckerman; pp. 467–487



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

The focus of the current paper is the slightly more general proof system *polynomial calculus resolution (PCR)*. This proof system was introduced by Clegg et al. [9] in a slightly weaker form that is usually referred to as *polynomial calculus (PC)* and was later extended by Alekhovich et al. [1]. In PC and PCR clauses are translated to multilinear polynomials over some (fixed) field  $\mathbb{F}$ , and a CNF formula  $F$  is shown to be unsatisfiable by proving that the constant 1 lies in the ideal generated by the polynomials corresponding to the clauses of  $F$ . Here the size of a proof is measured as the number of monomials in a proof when all polynomials are expanded out as linear combinations of monomials, and the width of a clause corresponds to the (total) *degree* of the polynomial representing the clause. Briefly, the difference between PC and PCR is that the latter proof system has separate formal variables for positive and negative literals over the same variable. Thanks to this, one can encode wide clauses into polynomials compactly regardless of the sign of the literals in the clauses, which allows PCR to simulate resolution efficiently. With respect to the degree measure PC and PCR are exactly the same, and furthermore the degree needed to prove in polynomial calculus that a formula is unsatisfiable is at most the width required in resolution.

In a work that served, interestingly enough, as a precursor to [5], Impagliazzo et al. [16] showed that strong lower bounds on the degree of PC proofs are sufficient to establish strong size lower bounds. The same proof goes through for PCR, and hence any lower bound on proof size obtained via a degree lower bound applies to both PC and PCR. In this paper, we will therefore be somewhat sloppy in distinguishing the two proof systems, sometimes writing “polynomial calculus” to refer to both systems when the results apply to both PC and PCR.

In contrast to the situation for resolution after [5], the paper [16] has not been followed by a corresponding development of a generally applicable machinery for proving degree lower bounds. For fields of characteristic distinct from 2 it is sometimes possible to obtain lower bounds by doing an affine transformation from  $\{0, 1\}$  to the “Fourier basis”  $\{-1, +1\}$ , an idea that seems to have appeared first in [7, 13]. For fields of arbitrary characteristic Alekhovich and Razborov [2] developed a powerful technique for general systems of polynomial equations, which when restricted to the standard encoding of CNF formulas  $F$  yields that polynomial calculus proofs require high degree if the corresponding bipartite clause-variable incidence graphs  $G(F)$  are good enough expanders. There are many formula families for which this is not true, however. One can have a constraint satisfaction problem where the constraint-variable incidence graph is an expander – say, for instance, for an unsatisfiable set of linear equations mod 2 – but where each constraint is then translated into several clauses when encoded into CNF, meaning that the clause-variable incidence graph  $G(F)$  will no longer be expanding. For some formulas this limitation is inherent – it is not hard to see that an inconsistent system of linear equations mod 2 is easy to refute in polynomial calculus over  $\mathbb{F}_2$  – but in other cases it would seem that some kind of expansion of this sort should still be enough, “morally speaking,” to guarantee that the CNF formulas are hard.

One important direction in proof complexity, which is the reason research in this area was initiated by Cook and Reckhow [10], has been to prove superpolynomial lower bounds on proof size for increasingly stronger proof systems. For proof systems where such lower bounds have already been obtained, however, a somewhat orthogonal research direction has been to try to gain a better understanding of the strengths and weaknesses of the proof system by studying different combinatorial principles (encoded in CNF) and determining how hard they are to prove.

It seems fair to say that by far the most extensively studied such combinatorial principle is the *pigeonhole principle*. This principle is encoded into CNF as unsatisfiable formulas claiming that  $m$  pigeons can be mapped in a one-to-one fashion into  $n$  holes for  $m > n$ , but

there are several choices exactly how to do this encoding. The most basic *pigeonhole principle (PHP) formulas* have clauses saying that every pigeon gets at least one pigeonhole and that no hole contains two pigeons. While these formulas are already unsatisfiable for  $m \geq n + 1$ , they do not a priori rule out “fat” pigeons residing in several holes. The *functional pigeonhole principle (FPHP) formulas* perhaps correspond more closely to our intuitive understanding of the pigeonhole principle in that they also contain *functionality* clauses specifying that every pigeon gets exactly one pigeonhole and not more. Another way of making the basic PHP formulas more constrained is to add *onto* clauses requiring that every pigeonhole should get a pigeon, yielding so-called *onto-PHP formulas*. Finally, the most restrictive encoding, and hence the hardest one when it comes to proving lower bounds, are the *onto-FPHP formulas* containing both functionality and onto clauses, i.e., saying that the mapping from pigeons to pigeonholes is a perfect matching. Razborov’s survey [23] gives a detailed account of these different flavours of the pigeonhole principle formulas and results for them with respect to various proof systems – we just quickly highlight some facts relevant to this paper below.

For the resolution proof system there is not much need to distinguish between the different PHP versions discussed above. The lower bound by Haken [14] for formulas with  $m = n + 1$  pigeons can be made to work also for onto-FPHP formulas, and more recent works by Raz [20] and Razborov [24, 25] show that the formulas remain exponentially hard (measured in the number of pigeonholes  $n$ ) even for arbitrarily many pigeons  $m$ .

Interestingly enough, for polynomial calculus the story is very different. The first degree lower bounds were proven by Razborov [21], but for a different encoding than the standard translation from CNF, since translating wide clauses yields initial polynomials of high degree. Alekhovich and Razborov [2] proved lower bounds for a 3-CNF version of the pigeonhole principle, from which it follows that the standard CNF encoding requires proofs of exponential size. However, as shown by Riis [27] the onto-FPHP formulas with  $m = n + 1$  pigeons are easy for polynomial calculus. And while the encoding in [21] also captures the functionality restriction in some sense, it has remained open whether the standard CNF encoding of functional pigeonhole principle formulas translated to polynomials is hard (this question has been highlighted, for instance, in Razborov’s open problems list [26]).

Another way of modifying the pigeonhole principle is to restrict the choices of pigeonholes for each pigeon by defining the formulas over a bipartite graph  $H = (U \cup V, E)$  with  $|U| = m$  and  $|V| = n$  and requiring that each pigeon  $u \in U$  goes to one of its neighbouring holes in  $N(u) \subseteq V$ . If the graph  $H$  has constant left degree, the corresponding *graph pigeonhole principle formula* has constant width and a linear number of variables, which makes it possible to apply [5, 16] to obtain exponential proof size lower bounds from linear width/degree lower bounds. A careful reading of the proofs in [2] reveals that this paper establishes linear polynomial calculus degree lower bounds (and hence exponential size lower bounds) for graph PHP formulas, and in fact also graph Onto-PHP formulas, over constant-degree expanders  $H$ . Razborov lists as one of the open problems in [23] whether this holds also for graph FPHP formulas, i.e., with functionality clauses added, from which exponential lower bounds on polynomial calculus proof size for the general FPHP formulas would immediately follow.

## 1.1 Our Results

We revisit the technique developed in [2] for proving polynomial calculus degree lower bounds, restricting our attention to the special case when the polynomials are obtained by the canonical translation of CNF formulas.

Instead of considering the standard bipartite clause-variable incidence graph  $G(F)$  of a CNF formula  $F$  (with clauses on the left, variables on the right, and edges encoding that a

variable occurs in a clause) we construct a new graph  $G'$  by clustering several clauses and/or variables into single vertices, reflecting the structure of the encoded combinatorial principle. The edges in this new graph  $G'$  are the ones induced by the original graph  $G(F)$  in the natural way, i.e., there is an edge from a left cluster to a right cluster in  $G'$  if any clause in the left cluster has an edge to any variable in the right cluster in  $G(F)$ . We remark that such a clustering is already implicit in, for instance, the resolution lower bounds in [5] for Tseitin formulas (which is essentially just a special form of unsatisfiable linear equations) and graph PHP formulas, as well as in the graph PHP lower bound for polynomial calculus in [2].

We then show that if this clustering is done in the right way, the proofs in [2] still go through and yield strong polynomial calculus degree lower bounds when  $G'$  is a good enough expander.<sup>1</sup> It is clear that this cannot work in general – as already discussed above, any inconsistent system of linear equations mod 2 is easy to refute in polynomial calculus over  $\mathbb{F}_2$ , even though for a random instance of this problem the clauses encoding each linear equation can be clustered to yield an excellent expander  $G'$ . Very informally (and somewhat incorrectly) speaking, the clustering should be such that if a cluster of clauses  $F'$  on the left is a neighbour of a variable cluster  $V$  on the right, then there should exist an assignment  $\rho$  to  $V$  such that  $\rho$  satisfies all of  $F'$  and such that for the clauses outside of  $F'$  they are either satisfied by  $\rho$  or left completely untouched by  $\rho$ . Also, it turns out to be helpful not to insist that the clustering of variables on the right should be a partition, but that we should allow the same variable to appear in several clusters if needed (as long as the number of clusters for each variable is bounded).

This extension of the lower bound method in [2] makes it possible to present previously obtained polynomial calculus degree lower bounds in [2, 12, 17] in a unified framework. Moreover, it allows us to prove the following new results:

1. If a bipartite graph  $H = (U \dot{\cup} V, E)$  with  $|U| = m$  and  $|V| = n$  is a boundary expander (a.k.a. unique-neighbour expander), then the graph FPHP formula over  $H$  requires proofs of linear polynomial calculus degree, and hence exponential polynomial calculus size.
2. Since FPHP formulas can be turned into graph FPHP formulas by hitting them with a restriction, and since restrictions can only decrease proof size, it follows that FPHP formulas require proofs of exponential size in polynomial calculus.

This fills in the last missing pieces in our understanding of the different flavours of pigeonhole principle formulas with  $n + 1$  pigeons and  $n$  holes for polynomial calculus. Namely, while Onto-FPHP formulas are easy for polynomial calculus, both FPHP formulas and Onto-PHP formulas are hard even when restricted to expander graphs.

## 1.2 Organization of This Paper

After reviewing the necessary preliminaries in Section 2, we present our extension of the Alekhovich–Razborov method in Section 3. In Section 4, we show how this method can be used to rederive some previous polynomial calculus degree lower bounds as well as to obtain new degree and size lower bounds for functional (graph) PHP formulas. We conclude in Section 5 by discussing some possible directions for future research. We refer to the full-length version [18] of this paper for the details omitted in this extended abstract.

---

<sup>1</sup> For a certain twist of the definition of expander that we do not describe in full detail here in order to keep the discussion at an informal, intuitive level. The formal description is given in Section 3.1.



## 2 Preliminaries

Let us start by giving an overview of the relevant proof complexity background. This material is standard and we refer to, for instance, the survey [19] for more details.

A *literal* over a Boolean variable  $x$  is either the variable  $x$  itself (a *positive literal*) or its negation  $\neg x$  or  $\bar{x}$  (a *negative literal*). We define  $\bar{\bar{x}} = x$ . We identify 0 with true and 1 with false. We remark that this is the opposite of the standard convention in proof complexity, but it is a more natural choice in the context of polynomial calculus, where “evaluating to true” means “vanishing.” A *clause*  $C = a_1 \vee \dots \vee a_k$  is a disjunction of literals. A *CNF formula*  $F = C_1 \wedge \dots \wedge C_m$  is a conjunction of clauses. The *width*  $W(C)$  of a clause  $C$  is the number of literals  $|C|$  in it, and the width  $W(F)$  of the formula  $F$  is the maximum width of any clause in the formula. We think of clauses and CNF formulas as sets, so that order is irrelevant and there are no repetitions. A  $k$ -CNF formula has all clauses of size at most  $k$ , where  $k$  is assumed to be some fixed constant.

In polynomial calculus resolution the goal is to prove the unsatisfiability of a CNF formula by reasoning with polynomials from a polynomial ring  $\mathbb{F}[x, \bar{x}, y, \bar{y}, \dots]$  (where  $x$  and  $\bar{x}$  are viewed as distinct formal variables) over some fixed field  $\mathbb{F}$ . The results in this paper hold for all fields  $\mathbb{F}$  regardless of characteristic. In what follows, a *monomial*  $m$  is a product of variables and a *term*  $t$  is a monomial multiplied by an arbitrary non-zero field element.

► **Definition 2.1** (Polynomial calculus resolution (PCR) [1, 9]). A *polynomial calculus resolution (PCR) refutation*  $\pi : F \vdash \perp$  of a CNF formula  $F$  (also referred to as a *PCR proof* for  $F$ ) over a field  $\mathbb{F}$  is an ordered sequence of polynomials  $\pi = (P_1, \dots, P_\tau)$ , expanded out as linear combinations of monomials, such that  $P_\tau = 1$  and each line  $P_i$ ,  $1 \leq i \leq \tau$ , is either

- a monomial  $\prod_{x \in L^+} x \cdot \prod_{y \in L^-} \bar{y}$  encoding a clause  $\bigvee_{x \in L^+} x \vee \bigvee_{y \in L^-} \bar{y}$  in  $F$  (a *clause axiom*);
- a *Boolean axiom*  $x^2 - x$  or *complementarity axiom*  $x + \bar{x} - 1$  for any variable  $x$ ;
- a polynomial obtained from one or two previous polynomials by *linear combination*  $\frac{Q - R}{\alpha Q + \beta R}$  or *multiplication*  $\frac{Q}{xQ}$  for any  $\alpha, \beta \in \mathbb{F}$  and any variable  $x$ .

If we drop complementarity axioms and encode each negative literal  $\bar{x}$  as  $(1 - x)$ , the proof system is called *polynomial calculus (PC)*.

The *size*  $S(\pi)$  of a PC/PCR refutation  $\pi = (P_1, \dots, P_\tau)$  is the number of monomials in  $\pi$  (counted with repetitions),<sup>2</sup> the *degree*  $Deg(\pi)$  is the maximal degree of any monomial appearing in  $\pi$ , and the *length*  $L(\pi)$  is the number  $\tau$  of polynomials in  $\pi$ . Taking the minimum over all PCR refutations of a formula  $F$ , we define the size  $S_{PCR}(F \vdash \perp)$ , degree  $Deg_{PCR}(F \vdash \perp)$ , and length  $L_{PCR}(F \vdash \perp)$  of refuting  $F$  in PCR (and analogously for PC).

We write  $Vars(C)$  and  $Vars(m)$  to denote the set of all variables appearing in a clause  $C$  or monomial (or term)  $m$ , respectively and extend this notation to CNF formulas and polynomials by taking unions. We use the notation  $\langle P_1, \dots, P_m \rangle$  for the ideal generated by the polynomials  $P_i$ ,  $i \in [m]$ . That is,  $\langle P_1, \dots, P_m \rangle$  is the minimal subset of polynomials containing all  $P_i$  that is closed under addition and multiplication by any polynomial. One way of viewing a polynomial calculus (PC or PCR) refutation is as a calculation in the ideal generated by the encodings of clauses in  $F$  and the Boolean and complementarity axioms. It can be shown that such an ideal contains 1 if and only if  $F$  is unsatisfiable.

<sup>2</sup> We remark that the natural definition of size is to count monomials with repetition, but all lower bound techniques known actually establish slightly stronger lower bounds on the number of *distinct* monomials.

As mentioned above, we have  $\text{Deg}_{\text{PCR}}(F \vdash \perp) = \text{Deg}_{\text{PC}}(F \vdash \perp)$  for any CNF formula  $F$ . This claim can essentially be verified by taking any PCR refutation of  $F$  and replacing all occurrences of  $\bar{y}$  by  $(1 - y)$  to obtain a valid PC refutation in the same degree. Hence, we can drop the subscript from the notation for the degree measure. We have the following relation between refutation size and refutation degree (which was originally proven for PC but the proof of which also works for PCR).

► **Theorem 2.2** ([16]). *Let  $F$  be an unsatisfiable CNF formula of width  $W(F)$  over  $n$  variables. Then*

$$S_{\text{PCR}}(F \vdash \perp) = \exp \left( \Omega \left( \frac{(\text{Deg}(F \vdash \perp) - W(F))^2}{n} \right) \right).$$

Thus, for  $k$ -CNF formulas it is sufficient to prove strong enough lower bounds on the PC degree of refutations to establish strong lower bounds on PCR proof size.

Furthermore, it will be convenient for us to simplify the definition of PC so that axioms  $x^2 - x$  are always applied implicitly whenever possible. We do this by defining the result of the multiplication operation to be the multilinearized version of the product. This can only decrease the degree (and size) of the refutation, and is in fact how polynomial calculus is defined in [2]. Hence, from now on whenever we refer to polynomials and monomials we mean multilinear polynomials and multilinear monomials, respectively, and polynomial calculus is defined over the (multilinear) polynomial ring  $\mathbb{F}[x, y, z, \dots] / \langle x^2 - x, y^2 - y, z^2 - z, \dots \rangle$ .

We will also need to use restrictions. A *restriction*  $\rho$  on  $F$  is a partial assignment to the variables of  $F$ . We use  $\text{Dom}(\rho)$  to denote the set of variables assigned by  $\rho$ . In a restricted formula  $F|_{\rho}$  all clauses satisfied by  $\rho$  are removed and all other clauses have falsified literals removed. For a PC refutation  $\pi$  restricted by  $\rho$  we have that if  $\rho$  satisfies a literal in a monomial, then that monomial is set to 0 and vanishes, and all falsified literals in a monomial get replaced by 1 and disappear. It is not hard to see that if  $\pi$  is a PC (or PCR) refutation of  $F$ , then  $\pi|_{\rho}$  is a PC (or PCR) refutation of  $F|_{\rho}$ , and this restricted refutation has at most the same size, degree, and length as the original refutation.

### 3 A Generalization of the Alekhovich–Razborov Method for CNFs

Many lower bounds in proof complexity are proved by arguing in terms of expansion. One common approach is to associate a bipartite graph  $G(F)$  with the CNF formula  $F$  with clauses on one side and variables on the other and with edges encoding that a variable occurs in a clause (the so-called *clause-variable incidence graph* mentioned in the introduction). The method we present below, which is an extension of the techniques developed by Alekhovich and Razborov [2] (but restricted to the special case of CNF formulas), is a variation on this theme. As already discussed, however, we will need a slightly more general graph construction where clauses and variables can be grouped into clusters. We begin by describing this construction.

#### 3.1 A Generalized Clause-Variable Incidence Graph

The key to our construction of generalized clause-variable incidence graphs is to keep track of how clauses in a CNF formula are affected by partial assignments.

► **Definition 3.1** (Respectful assignments and variable sets). We say that a partial assignment  $\rho$  *respects* a CNF formula  $E$ , or that  $\rho$  is  *$E$ -respectful*, if for every clause  $C$  in  $E$  either  $\text{Vars}(C) \cap \text{Dom}(\rho) = \emptyset$  or  $\rho$  satisfies  $C$ . A set of variables  $V$  respects a CNF formula  $E$  if there exists an assignment  $\rho$  with  $\text{Dom}(\rho) = V$  that respects  $E$ .

► **Definition 3.2** (Respectful satisfaction). Let  $F$  and  $E$  be CNF formulas and let  $V$  be a set of variables. We say that  $F$  is  $E$ -respectfully satisfiable by  $V$  if there exists a partial assignment  $\rho$  with  $\text{Dom}(\rho) = V$  that satisfies  $F$  and respects  $E$ . Such an assignment  $\rho$  is said to  $E$ -respectfully satisfy  $F$ .

Using a different terminology, Definition 3.1 says that  $\rho$  is an *autarky* for  $E$ , meaning that  $\rho$  satisfies all clauses in  $E$  which it touches, i.e., that  $E|_{\rho} \subseteq E$  after we remove all satisfied clauses in  $E|_{\rho}$ . Definition 3.2 ensures that the autarky  $\rho$  satisfies the formula  $F$ .

Recall that we identify a CNF formula  $\bigwedge_{i=1}^m C_i$  with the set of clauses  $\{C_i \mid i \in [m]\}$ . In the rest of this section we will switch freely between these two perspectives. We also change to the notation  $\mathcal{F}$  for the input CNF formula, to free up other letters that will be needed in notation introduced below.

To build a bipartite graph representing the CNF formula  $\mathcal{F}$ , we will group the formula into subformulas (i.e., subsets of clauses). In what follows, we write  $\mathcal{U}$  to denote the part of  $\mathcal{F}$  that will form the left vertices of the constructed bipartite graph, while  $\mathcal{V}$  denotes the part of  $\mathcal{F}$  which will not be represented in the graph but will be used to enforce respectful satisfaction. In more detail,  $\mathcal{U}$  is a family of subformulas  $F$  of  $\mathcal{F}$  where each subformula is one vertex on the left-hand side of the graph. We also consider the variables of  $\mathcal{F}$  to be divided into a family  $\mathcal{V}$  of subsets of variables  $V$ . In our definition,  $\mathcal{U}$  and  $\mathcal{V}$  do not need to be partitions of clauses and variables in  $\mathcal{F}$ , respectively. This is not too relevant for  $\mathcal{U}$  because we will always define it as a partition, but it turns out to be useful in our applications to have sets in  $\mathcal{V}$  share variables. The next definition describes the bipartite graph that we build and distinguishes between two types of neighbour relations in this graph.

► **Definition 3.3** (Bipartite  $(\mathcal{U}, \mathcal{V})_E$ -graph). Let  $E$  be a CNF formula,  $\mathcal{U}$  be a set of CNF formulas, and  $\mathcal{V}$  be a family of sets of variables  $V$  that respect  $E$ . Then the (*bipartite*)  $(\mathcal{U}, \mathcal{V})_E$ -graph is a bipartite graph with left vertices  $F \in \mathcal{U}$ , right vertices  $V \in \mathcal{V}$ , and edges between  $F$  and  $V$  if  $\text{Vars}(F) \cap V \neq \emptyset$ . For every edge  $(F, V)$  in the graph we say that  $F$  and  $V$  are  $E$ -respectful neighbours if  $F$  is  $E$ -respectfully satisfiable by  $V$ . Otherwise, they are  $E$ -disrespectful neighbours.

We will often write  $(\mathcal{U}, \mathcal{V})_E$  as a shorthand for the graph defined by  $\mathcal{U}$ ,  $\mathcal{V}$ , and  $E$  as above. We will also use standard graph notation and write  $N(F)$  to denote the set of all neighbours  $V \in \mathcal{V}$  of a vertex/CNF formula  $F \in \mathcal{U}$ . It is important to note that the fact that  $F$  and  $V$  are  $E$ -respectful neighbours can be witnessed by an assignment that falsifies other subformulas  $F' \in \mathcal{U} \setminus \{F\}$ .

► **Definition 3.4** (Respectful boundary). For a  $(\mathcal{U}, \mathcal{V})_E$ -graph and a subset  $\mathcal{U}' \subseteq \mathcal{U}$ , the  $E$ -respectful boundary  $\partial_E(\mathcal{U}')$  of  $\mathcal{U}'$  is the family of variable sets  $V \in \mathcal{V}$  such that each  $V \in \partial_E(\mathcal{U}')$  is an  $E$ -respectful neighbour of some clause set  $F \in \mathcal{U}'$  but is not a neighbour (respectful or disrespectful) of any other clause set  $F' \in \mathcal{U}' \setminus \{F\}$ .

It will sometimes be convenient to interpret subsets  $\mathcal{U}' \subseteq \mathcal{U}$  as formulas  $\bigwedge_{F \in \mathcal{U}'} \bigwedge_{C \in F} C$ , and we will switch back and forth between these two interpretations as seems most suitable. We will show that a formula  $\mathcal{F} = \bigwedge_{F \in \mathcal{U}} \bigwedge_{C \in F} C \wedge E = \mathcal{U} \wedge E$  is hard for polynomial calculus with respect to degree if the  $(\mathcal{U}, \mathcal{V})_E$ -graph has a certain expansion property as defined next.

► **Definition 3.5** (Respectful boundary expander). A  $(\mathcal{U}, \mathcal{V})_E$ -graph is said to be an  $(s, \delta, \xi, E)$ -respectful boundary expander, or just an  $(s, \delta, \xi, E)$ -expander for brevity, if for every set  $\mathcal{U}' \subseteq \mathcal{U}$ ,  $|\mathcal{U}'| \leq s$ , it holds that  $|\partial_E(\mathcal{U}')| \geq \delta|\mathcal{U}'| - \xi$ .

Before we state our main theorem we need one more technical definition, which is used to ensure that there do not exist variables that appear in too many variable sets in  $\mathcal{V}$ .

► **Definition 3.6.** The *overlap* of a variable  $x$  with respect to a family of variable sets  $\mathcal{V}$  is  $ol(x, \mathcal{V}) = |\{V \in \mathcal{V} : x \in V\}|$  and the overlap of  $\mathcal{V}$  is  $ol(\mathcal{V}) = \max_x \{ol(x, \mathcal{V})\}$ , i.e., the maximum number of sets  $V \in \mathcal{V}$  containing any particular variable  $x$ .

Given the above definitions, we can state the main technical result in this paper as follows.

► **Theorem 3.7.** Let  $\mathcal{F} = \bigwedge_{F \in \mathcal{U}} \bigwedge_{C \in \mathcal{F}} C \wedge E = \mathcal{U} \wedge E$  be a CNF formula for which  $(\mathcal{U}, \mathcal{V})_E$  is an  $(s, \delta, \xi, E)$ -expander with overlap  $ol(\mathcal{V}) = d$ , and suppose furthermore that for all  $\mathcal{U}' \subseteq \mathcal{U}$ ,  $|\mathcal{U}'| \leq s$ , it holds that  $\mathcal{U}' \wedge E$  is satisfiable. Then any polynomial calculus refutation of  $\mathcal{F}$  requires degree strictly greater than  $(\delta s - 2\xi)/(2d)$ .

In order to prove this theorem, it will be convenient to review some algebra. We do so next.

### 3.2 Some Algebra Basics

We will need to compute with polynomials modulo ideals, and in order to do so we need to have an ordering of monomials (which, as we recall, will always be multilinear).

► **Definition 3.8 (Admissible ordering).** We say that a total ordering  $\prec$  on the set of all monomials over some fixed set of variables is *admissible* if the following conditions hold:

- If  $Deg(m_1) < Deg(m_2)$ , then  $m_1 \prec m_2$ .
- For any  $m_1, m_2$ , and  $m$  such that  $m_1 \prec m_2$  and  $Vars(m) \cap (Vars(m_1) \cup Vars(m_2)) = \emptyset$ , it holds that  $mm_1 \prec mm_2$ .

Two terms  $t_1 = \alpha_1 m_1$  and  $t_2 = \alpha_2 m_2$  are ordered in the same way as their underlying monomials  $m_1$  and  $m_2$ .

One example of an admissible ordering is to first order monomials with respect to their degree and then lexicographically. We write  $m_1 \preceq m_2$  to denote that  $m_1 \prec m_2$  or  $m_1 = m_2$ .

► **Definition 3.9 (Leading, reducible, and irreducible terms).** For a polynomial  $P = \sum_i t_i$ , the *leading term*  $LT(P)$  of  $P$  is the largest term  $t_i$  according to  $\prec$ . Let  $I$  be an ideal over the (multilinear) polynomial ring  $\mathbb{F}[x, y, z, \dots]/\langle x^2 - x, y^2 - y, z^2 - z, \dots \rangle$ . We say that a term  $t$  is *reducible modulo*  $I$  if there exists a polynomial  $Q \in I$  such that  $t = LT(Q)$  and that  $t$  is *irreducible modulo*  $I$  otherwise.

The following fact is not hard to verify.

► **Fact 3.10.** Let  $I$  be an ideal over  $\mathbb{F}[x, y, z, \dots]/\langle x^2 - x, y^2 - y, z^2 - z, \dots \rangle$ . Then any multilinear polynomial  $P \in \mathbb{F}[x, y, z, \dots]/\langle x^2 - x, y^2 - y, z^2 - z, \dots \rangle$  can be written uniquely as a sum  $Q + R$ , where  $Q \in I$  and  $R$  is a linear combination of irreducible terms modulo  $I$ .

This is what allows us to reduce polynomials modulo an ideal in a well-defined manner.

► **Definition 3.11 (Reduction operator).** Let  $I$  be an ideal and let  $P$  be any multilinear polynomial over  $\mathbb{F}[x, y, z, \dots]/\langle x^2 - x, y^2 - y, z^2 - z, \dots \rangle$ . The *reduction operator*  $R_I$  is the operator that when applied to  $P$  returns the sum of irreducible terms  $R_I(P) = R$  such that  $P - R \in I$ .

We conclude our brief algebra review by stating two observations that are more or less immediate, but are helpful enough for us to want to highlight them explicitly.

► **Observation 3.12.** For any two ideals  $I_1, I_2$  such that  $I_1 \subseteq I_2$  and any two polynomials  $P, P'$  it holds that  $R_{I_2}(P \cdot R_{I_1}(P')) = R_{I_2}(PP')$ .

► **Observation 3.13.** Suppose that the term  $t$  is irreducible modulo the ideal  $I$  and let  $\rho$  be any partial assignment of variables in  $Vars(t)$  to values in  $\mathbb{F}$  such that  $t|_\rho \neq 0$ . Then  $t|_\rho$  is also irreducible modulo  $I$ .

### 3.3 Proof Strategy

Let us now state the lemma on which we base the proof of Theorem 3.7.

► **Lemma 3.14** ([21]). *Let  $\mathcal{F}$  be any CNF formula and  $D \in \mathbb{N}^+$  be a positive integer. Suppose that there exists a linear operator  $R$  on multilinear polynomials over  $\text{Vars}(\mathcal{F})$  with the following properties:*

1.  $R(1) \neq 0$ .
2.  $R(C) = 0$  for (the translations to polynomials of) all axioms  $C \in \mathcal{F}$ .
3. For every term  $t$  with  $\text{Deg}(t) < D$  and every variable  $x$  it holds that  $R(xt) = R(xR(t))$ .

*Then any polynomial calculus refutation of  $\mathcal{F}$  (and hence any PCR refutation of  $\mathcal{F}$ ) requires degree strictly greater than  $D$ .*

To prove Theorem 3.7, we construct a linear operator  $R_{\mathcal{G}}$  that satisfies the conditions of Lemma 3.14 when the  $(\mathcal{U}, \mathcal{V})_E$ -graph  $\mathcal{G}$  is an expander. First, let us describe how we make the connection between polynomials and the given  $(\mathcal{U}, \mathcal{V})_E$ -graph. We remark that in the rest of this section we will identify a clause  $C$  with its polynomial translation and will refer to  $C$  as a (polynomial) axiom.

► **Definition 3.15** (Term and polynomial neighbourhood). The *neighbourhood*  $N(t)$  of a term  $t$  with respect to  $(\mathcal{U}, \mathcal{V})_E$  is  $N(t) = \{V \in \mathcal{V} \mid \text{Vars}(t) \cap V \neq \emptyset\}$ , i.e., the family of all variable sets containing variables mentioned by  $t$ . The neighbourhood of a polynomial  $P = \sum_i t_i$  is  $N(P) = \bigcup_i N(t_i)$ , i.e., the union of the neighbourhoods of all terms in  $P$ .

To every polynomial we can now assign a family of variable sets  $\mathcal{V}'$ . But we are interested in the axioms that are needed in order to produce that polynomial. That is, given a family of variable sets  $\mathcal{V}'$ , we would like to identify the largest set of axioms  $\mathcal{U}'$  that could possibly have been used in a derivation that yielded polynomials  $P$  with  $\text{Vars}(P) \subseteq \bigcup_{V \in \mathcal{V}'} V$ . This is the intuition behind the next definition.<sup>3</sup>

► **Definition 3.16** (Polynomial support). For a given  $(\mathcal{U}, \mathcal{V})_E$ -graph and a family of variable sets  $\mathcal{V}' \subseteq \mathcal{V}$ , we say that a subset  $\mathcal{U}' \subseteq \mathcal{U}$  is  $(s, \mathcal{V}')$ -*contained* if  $|\mathcal{U}'| \leq s$  and  $\partial_E(\mathcal{U}') \subseteq \mathcal{V}'$ .

We define the *polynomial  $s$ -support*  $\text{Sup}_s(\mathcal{V}')$  of  $\mathcal{V}'$  with respect to  $(\mathcal{U}, \mathcal{V})_E$ , or just  *$s$ -support* of  $\mathcal{V}'$  for brevity, to be the union of all  $(s, \mathcal{V}')$ -contained subsets  $\mathcal{U}' \subseteq \mathcal{U}$ , and the  $s$ -support  $\text{Sup}_s(t)$  of a term  $t$  is defined to be the  $s$ -support of  $N(t)$ .

We will usually just speak about “support” below without further qualifying this term, since the  $(\mathcal{U}, \mathcal{V})_E$ -graph  $\mathcal{G}$  will be clear from context. The next observation follows immediately from Definition 3.16.

► **Observation 3.17.** *Support is monotone in the sense that if  $t \subseteq t'$  are two terms, then it holds that  $\text{Sup}_s(t) \subseteq \text{Sup}_s(t')$ .*

Once we have identified the axioms that are potentially involved in deriving  $P$ , we define the linear operator  $R_{\mathcal{G}}$  as the reduction modulo the ideal generated by these axioms as in Definition 3.11. We will show that under the assumptions in Theorem 3.7 it holds that this operator satisfies the conditions in Lemma 3.14. Let us first introduce some notation for the set of all polynomials that can be generated from some axioms  $\mathcal{U}' \subseteq \mathcal{U}$ .

<sup>3</sup> We remark that Definition 3.16 is a slight modification of the original definition of support in [2] that was proposed by Yuval Filmus [11].

► **Definition 3.18.** For a  $(\mathcal{U}, \mathcal{V})_E$ -graph and  $\mathcal{U}' \subseteq \mathcal{U}$ , we write  $\mathcal{I}_E(\mathcal{U}')$  to denote the ideal generated by the polynomial axioms in  $\mathcal{U}' \wedge E$ .<sup>4</sup>

► **Definition 3.19** ( $(\mathcal{U}, \mathcal{V})_E$ -graph reduction). For a  $(\mathcal{U}, \mathcal{V})_E$ -graph  $\mathcal{G}$ , the  $(\mathcal{U}, \mathcal{V})_E$ -graph reduction  $R_{\mathcal{G}}$  on a term  $t$  is defined as  $R_{\mathcal{G}}(t) = R_{\mathcal{I}_E(\text{Sup}_s(t))}(t)$ . For a polynomial  $P$ , we define  $R_{\mathcal{G}}(P)$  to be the linear extension of the operator  $R_{\mathcal{G}}$  defined on terms.

### 3.4 Some Properties of Polynomial Support

A crucial technical property that we will need is that if a  $(\mathcal{U}, \mathcal{V})_E$ -graph is a good expander in the sense of Definition 3.5, then for small enough sets  $\mathcal{V}'$  all  $(s, \mathcal{V}')$ -contained subsets  $\mathcal{U}' \subseteq \mathcal{U}$  as per Definition 3.16 are of at most half of the allowed size.

► **Lemma 3.20.** *Let  $(\mathcal{U}, \mathcal{V})_E$  be an  $(s, \delta, \xi, E)$ -expander and let  $\mathcal{V}' \subseteq \mathcal{V}$  be such that  $|\mathcal{V}'| \leq \delta s/2 - \xi$ . Then it holds that every  $(s, \mathcal{V}')$ -contained subset  $\mathcal{U}' \subseteq \mathcal{U}$  is in fact  $(s/2, \mathcal{V}')$ -contained.*

**Proof.** As  $|\mathcal{U}'| \leq s$  we can appeal to the expansion property of the  $(\mathcal{U}, \mathcal{V})_E$ -graph to derive the inequality  $|\partial_E(\mathcal{U}')| \geq \delta|\mathcal{U}'| - \xi$ . In the other direction, we can obtain an upper bound on the size of  $\partial_E(\mathcal{U}')$  by noting that for any  $(s, \mathcal{V}')$ -contained set  $\mathcal{U}'$  it holds that  $|\partial_E(\mathcal{U}')| \leq |\mathcal{V}'|$ . If we combine these bounds and use the assumption that  $|\mathcal{V}'| \leq \delta s/2 - \xi$ , we can conclude that  $|\mathcal{U}'| \leq s/2$ , which proves that  $\mathcal{U}'$  is  $(s/2, \mathcal{V}')$ -contained. ◀

Even more importantly, Lemma 3.20 now allows us to conclude that for a small enough subset  $\mathcal{V}'$  on the right-hand side of  $(\mathcal{U}, \mathcal{V})_E$  it holds that in fact the whole polynomial  $s$ -support  $\text{Sup}_s(\mathcal{V}')$  of  $\mathcal{V}'$  on the left-hand side is  $(s/2, \mathcal{V}')$ -contained.

► **Lemma 3.21.** *Let  $(\mathcal{U}, \mathcal{V})_E$  be an  $(s, \delta, \xi, E)$ -expander and let  $\mathcal{V}' \subseteq \mathcal{V}$  be such that  $|\mathcal{V}'| \leq \delta s/2 - \xi$ . Then the  $s$ -support  $\text{Sup}_s(\mathcal{V}')$  of  $\mathcal{V}'$  with respect to  $(\mathcal{U}, \mathcal{V})_E$  is  $(s/2, \mathcal{V}')$ -contained.*

**Proof.** We show that for any pair of  $(s, \mathcal{V}')$ -contained sets  $\mathcal{U}_1, \mathcal{U}_2 \subseteq \mathcal{U}$  their union  $\mathcal{U}_1 \cup \mathcal{U}_2$  is also  $(s, \mathcal{V}')$ -contained. First, by Lemma 3.20 we have  $|\mathcal{U}_1|, |\mathcal{U}_2| \leq s/2$  and hence  $|\mathcal{U}_1 \cup \mathcal{U}_2| \leq s$ . Second, it holds that  $\partial_E(\mathcal{U}_1), \partial_E(\mathcal{U}_2) \subseteq \mathcal{V}'$ , which implies that  $\partial_E(\mathcal{U}_1 \cup \mathcal{U}_2) \subseteq \mathcal{V}'$ , because taking the union of two sets can only shrink the boundary. This establishes that  $\mathcal{U}_1 \cup \mathcal{U}_2$  is  $(s, \mathcal{V}')$ -contained.

By induction on the number of  $(s, \mathcal{V}')$ -contained sets we can conclude that the support  $\text{Sup}_s(\mathcal{V}')$  is  $(s, \mathcal{V}')$ -contained as well, after which one final application of Lemma 3.20 shows that this set is  $(s/2, \mathcal{V}')$ -contained. This completes the proof. ◀

What the next lemma says is, roughly, that if we reduce a term  $t$  modulo an ideal generated by a not too large set of polynomials containing some polynomials outside of the support of  $t$ , then we can remove all such polynomials from the generators of the ideal without changing the irreducible component of  $t$ .

► **Lemma 3.22.** *Let  $\mathcal{G}$  be a  $(\mathcal{U}, \mathcal{V})_E$ -graph and let  $t$  be any term. Suppose that  $\mathcal{U}' \subseteq \mathcal{U}$  is such that  $\mathcal{U}' \supseteq \text{Sup}_s(t)$  and  $|\mathcal{U}'| \leq s$ . Then for any term  $t'$  with  $N(t') \subseteq N(\text{Sup}_s(t)) \cup N(t)$  it holds that if  $t'$  is reducible modulo  $\mathcal{I}_E(\mathcal{U}')$ , it is also reducible modulo  $\mathcal{I}_E(\text{Sup}_s(t))$ .*

<sup>4</sup> That is,  $\mathcal{I}_E(\mathcal{U}')$  is the smallest set  $I$  of multilinear polynomials that contains all axioms in  $\mathcal{U}' \wedge E$  and that is closed under addition of  $P_1, P_2 \in I$  and by multiplication of  $P \in I$  by any multilinear polynomial over  $\text{Vars}(\mathcal{U} \wedge E)$  (where as before the resulting product is implicitly multilinearized).



**Proof.** If  $\mathcal{U}'$  is  $(s, N(t))$ -contained, then by Definition 3.16 it holds that  $\mathcal{U}' \subseteq \text{Sup}_s(t)$  and there is nothing to prove. Hence, assume  $\mathcal{U}'$  is not  $(s, N(t))$ -contained. We claim that this implies that we can find a subformula  $F \in \mathcal{U}' \setminus \text{Sup}_s(t)$  with a neighbouring subset of variables  $V \in (\partial_E(\mathcal{U}') \cap N(F)) \setminus N(t')$  in the respectful boundary of  $\mathcal{U}'$  but not in the neighbourhood of  $t'$ . To argue this, note that since  $|\mathcal{U}'| \leq s$  it follows from Definition 3.16 that the reason  $\mathcal{U}'$  is not  $(s, N(t))$ -contained is that there exist some  $F \in \mathcal{U}'$  and some set of variables  $V \in N(F)$  such that  $V \in \partial_E(\mathcal{U}') \setminus N(t)$ . Moreover, the assumption  $\mathcal{U}' \supseteq \text{Sup}_s(t)$  implies that such an  $F$  cannot be in  $\text{Sup}_s(t)$ . Otherwise there would exist an  $(s, N(t))$ -contained set  $\mathcal{U}^*$  such that  $F \in \mathcal{U}^* \subseteq \text{Sup}_s(t) \subseteq \mathcal{U}'$ , from which it would follow that  $V \in \partial_E(\mathcal{U}') \cap N(\mathcal{U}^*) \subseteq \partial_E(\mathcal{U}^*) \subseteq N(t)$ , contradicting  $V \notin N(t)$ . We have shown that  $F \notin \text{Sup}_s(t) \subseteq \mathcal{U}'$  and  $V \in \partial_E(\mathcal{U}') \cap N(F)$ , and by combining these two facts we can also deduce that  $V \notin N(\text{Sup}_s(t))$ , since otherwise  $V$  could not be contained in the boundary of  $\mathcal{U}'$ . In particular, this means that  $V \notin N(t') \subseteq N(\text{Sup}_s(t)) \cup N(t)$ , which establishes the claim made above.

Fixing  $F$  and  $V$  such that  $F \in \mathcal{U}' \setminus \text{Sup}_s(t)$  and  $V \in (\partial_E(\mathcal{U}') \cap N(F)) \setminus N(t')$ , our second claim is that if  $F$  is removed from the generators of the ideal, it still holds that if  $t'$  is reducible modulo  $\mathcal{I}_E(\mathcal{U}')$ , then this term is also reducible modulo  $\mathcal{I}_E(\mathcal{U}' \setminus \{F\})$ . Given this second claim we are done, since we can then argue by induction over the elements in  $\mathcal{U}' \setminus \text{Sup}_s(t)$  and remove them one by one to arrive at the conclusion that every term  $t'$  with  $N(t') \subseteq N(\text{Sup}_s(t)) \cup N(t)$  that is reducible modulo  $\mathcal{I}_E(\mathcal{U}')$  is also reducible modulo  $\mathcal{I}_E(\text{Sup}_s(t))$ , which is precisely what the lemma says.

We proceed to establish this second claim. The assumption that  $t'$  is reducible modulo  $\mathcal{I}_E(\mathcal{U}')$  means that there exists a polynomial  $P \in \mathcal{I}_E(\mathcal{U}')$  such that  $t' = LT(P)$ . Since  $P$  is in the ideal  $\mathcal{I}_E(\mathcal{U}')$  it can be written as a polynomial combination  $P = \sum_i P_i C_i$  of axioms  $C_i \in \mathcal{U}' \wedge E$  for some polynomials  $P_i$ . If we could hit  $P$  with a restriction that satisfies (and hence removes)  $F$  while leaving  $t'$  and  $(\mathcal{U}' \setminus \{F\}) \wedge E$  untouched, this would show that  $t'$  is the leading term of some polynomial combination of axioms in  $(\mathcal{U}' \setminus \{F\}) \wedge E$ . This is almost what we are going to do.

As our restriction  $\rho$  we choose an arbitrary assignment with domain  $\text{Dom}(\rho) = V$  that  $E$ -respectfully satisfies  $F$ . Note that at least one such assignment exists since  $V \in \partial_E(\mathcal{U}') \cap N(F)$  is an  $E$ -respectful neighbour of  $F$  by Definition 3.4. By the choice of  $\rho$  it holds that  $F$  is satisfied, i.e., that all axioms in  $F$  are set to 0. Furthermore, none of the axioms in  $\mathcal{U}' \setminus \{F\}$  are affected by  $\rho$  since  $V$  is in the boundary of  $\mathcal{U}'$ .<sup>5</sup> As for axioms in  $E$  it is not necessarily true that  $\rho$  will leave all of them untouched, but by assumption  $\rho$  respects  $E$  and so any axiom in  $E$  is either satisfied (and zeroed out) by  $\rho$  or is left intact. It follows that  $P|_\rho$  can be written as a polynomial combination  $P|_\rho = \sum_i (P_i|_\rho) C_i$ , where  $C_i \in (\mathcal{U}' \setminus \{F\}) \wedge E$ , and hence  $P|_\rho \in \mathcal{I}_E(\mathcal{U}' \setminus \{F\})$ .

To see that  $t'$  is preserved as the leading term of  $P|_\rho$ , note that  $\rho$  does not assign any variables in  $t'$  since  $V \notin N(t')$ . Hence,  $t' = LT(P|_\rho)$ , as  $\rho$  can only make the other terms smaller with respect to  $\prec$ . This shows that there is a polynomial  $P' = P|_\rho \in \mathcal{I}_E(\mathcal{U}' \setminus \{F\})$  with  $LT(P') = t'$ , and hence  $t'$  is reducible modulo  $\mathcal{I}_E(\mathcal{U}' \setminus \{F\})$ . The lemma follows.  $\blacktriangleleft$

We need to deal with one more detail before we can prove the key technical lemma that it is possible to reduce modulo suitably chosen larger ideals without changing the reduction operator. We refer to the full-length version [18] for the proof of the next lemma.

<sup>5</sup> Recalling the remark after Definition 3.3, we note that we can ignore here if  $\rho$  happens to falsify axioms in  $\mathcal{U} \setminus \mathcal{U}'$ .



► **Lemma 3.23.** *Suppose that  $\mathcal{U}^* \subseteq \mathcal{U}$  for some  $(\mathcal{U}, \mathcal{V})_E$ -graph and let  $t$  be any term. Then it holds that  $N(R_{\mathcal{I}_E(\mathcal{U}^*)}(t)) \subseteq N(\mathcal{U}^*) \cup N(t)$ .*

Now we can state the formal claim that enlarging the ideal does not change the reduction operator if the enlargement is done in the right way.

► **Lemma 3.24.** *Let  $\mathcal{G}$  be a  $(\mathcal{U}, \mathcal{V})_E$ -graph and let  $t$  be any term. Suppose that  $\mathcal{U}' \subseteq \mathcal{U}$  is such that  $\mathcal{U}' \supseteq \text{Sup}_s(t)$  and  $|\mathcal{U}'| \leq s$ . Then it holds that  $R_{\mathcal{I}_E(\mathcal{U}')} (t) = R_{\mathcal{I}_E(\text{Sup}_s(t))} (t)$ .*

**Proof.** We prove  $R_{\mathcal{I}_E(\mathcal{U}')} (t) = R_{\mathcal{I}_E(\text{Sup}_s(t))} (t)$  by applying the contrapositive of Lemma 3.22. Recall that this lemma states that any term  $t'$  with  $N(t') \subseteq N(\text{Sup}_s(t)) \cup N(t)$  that is reducible modulo  $\mathcal{I}_E(\mathcal{U}')$  is also reducible modulo  $\mathcal{I}_E(\text{Sup}_s(t))$ . Since every term  $t'$  in  $R_{\mathcal{I}_E(\text{Sup}_s(t))} (t)$  is irreducible modulo  $\mathcal{I}_E(\text{Sup}_s(t))$  and since by applying Lemma 3.23 with  $\mathcal{U}^* = \text{Sup}_s(t)$  we have that  $N(t') \subseteq N(\text{Sup}_s(t)) \cup N(t)$ , it follows that  $t'$  is also irreducible modulo  $\mathcal{I}_E(\mathcal{U}')$ . This shows that  $R_{\mathcal{I}_E(\mathcal{U}')} (t) = R_{\mathcal{I}_E(\text{Sup}_s(t))} (t)$  as claimed, and the lemma follows. ◀

### 3.5 Putting the Pieces in the Proof Together

We just need two more lemmas to establish Theorem 3.7. To keep the length of this extended abstract reasonable, we just state these lemmas and hint at how to prove them.

► **Lemma 3.25.** *Let  $(\mathcal{U}, \mathcal{V})_E$  be an  $(s, \delta, \xi, E)$ -expander with overlap  $ol(\mathcal{V}) = d$ . Then for any term  $t$  with  $\text{Deg}(t) \leq (\delta s - 2\xi)/(2d)$  it holds that  $|\text{Sup}_s(t)| \leq s/2$ .*

This is a fairly straightforward application of Lemma 3.21.

► **Lemma 3.26.** *Let  $(\mathcal{U}, \mathcal{V})_E$  be an  $(s, \delta, \xi, E)$ -expander with overlap  $ol(\mathcal{V}) = d$ . Then for any term  $t$  with  $\text{Deg}(t) < \lfloor (\delta s - 2\xi)/(2d) \rfloor$ , any term  $t'$  occurring in  $R_{\mathcal{I}_E(\text{Sup}_s(t))} (t)$ , and any variable  $x$ , it holds that  $R_{\mathcal{I}_E(\text{Sup}_s(xt'))} (xt') = R_{\mathcal{I}_E(\text{Sup}_s(xt))} (xt')$ .*

This lemma follows from Observation 3.17, Lemma 3.23, Lemma 3.24, and Lemma 3.25.

**Proof of Theorem 3.7.** Recall that the assumptions of the theorem are that we have a  $(\mathcal{U}, \mathcal{V})_E$ -graph for a CNF formula  $\mathcal{F} = \bigwedge_{F \in \mathcal{U}} F \wedge E$  such that  $(\mathcal{U}, \mathcal{V})_E$  is an  $(s, \delta, \xi, E)$ -expander with overlap  $ol(\mathcal{V}) = d$  and that furthermore for all  $\mathcal{U}' \subseteq \mathcal{U}$ ,  $|\mathcal{U}'| \leq s$ , it holds that  $\bigwedge_{F \in \mathcal{U}'} F \wedge E$  is satisfiable. We want to prove that no polynomial calculus derivation from  $\bigwedge_{F \in \mathcal{U}} F \wedge E = \mathcal{U} \wedge E$  of degree at most  $(\delta s - 2\xi)/(2d)$  can reach contradiction.

We can focus on a  $(\mathcal{U}, \mathcal{V})_E$ -graph where the degree of axioms in  $\mathcal{U} \wedge E$  is at most  $(\delta s - 2\xi)/(2d)$ , as it is not hard to show that axioms of higher degree can safely be ignored. We want to show that the operator  $R_{\mathcal{G}}$  from Definition 3.19 satisfies the conditions of Lemma 3.14, from which Theorem 3.7 immediately follows. We can note right away that the operator  $R_{\mathcal{G}}$  is linear by construction.

To prove that  $R_{\mathcal{G}}(1) = R_{\mathcal{I}_E(\text{Sup}_s(1))}(1) \neq 0$ , we start by observing that the size of the  $s$ -support of 1 is upper-bounded by  $s/2$  according to Lemma 3.25. Using the assumption that for every subset  $\mathcal{U}'$  of  $\mathcal{U}$ ,  $|\mathcal{U}'| \leq s$ , the formula  $\mathcal{U}' \wedge E$  is satisfiable, it follows that 1 is not in the ideal  $\mathcal{I}_E(\text{Sup}_s(1))$  and hence  $R_{\mathcal{I}_E(\text{Sup}_s(1))}(1) \neq 0$ .

We next show that  $R_{\mathcal{G}}(C) = 0$  for any axiom clause  $C \in \mathcal{U} \wedge E$  (where we recall that we identify a clause  $C$  with its translation into a linear combination of monomials). By the assumption above it holds that the degree of  $C$  is bounded by  $(\delta s - 2\xi)/(2d)$ , from which it follows by Lemma 3.25 that the size of the  $s$ -support of every term in  $C$  is bounded by  $s/2$ . Since  $C$  is the polynomial encoding of a clause, the leading term  $LT(C)$  contains all the

variables appearing in  $C$ .<sup>6</sup> Hence, the  $s$ -support  $Sup_s(LT(C))$  of the leading term contains the  $s$ -support of every other term in  $C$  by Observation 3.17 and we can use Lemma 3.24 to conclude that  $R_G(C) = R_{\mathcal{I}_E(Sup_s(LT(C)))}(C)$ . If  $C \in E$ , this means we are done because  $\mathcal{I}_E(Sup_s(LT(C)))$  contains all of  $E$ , implying that  $R_G(C) = 0$ .

For  $C \in \mathcal{U}$  we cannot immediately argue that  $C$  reduces to 0, since (in contrast to [2]) it is not immediately clear that  $Sup_s(LT(C))$  contains  $C$ . The problem here is that we might worry that  $C$  is part of some subformula  $F \in \mathcal{U}$  for which the boundary  $\partial_E(F)$  is not contained in  $N(LT(C)) = Vars(C)$ , and hence there is no obvious reason why  $C$  should be a member of any  $(s, N(LT(C)))$ -contained subset of  $\mathcal{U}$ . However, in view of Lemma 3.24 (applied, strictly speaking, once for every term in  $C$ ) we can choose some  $F \in \mathcal{U}$  such that  $C \in F$  and add it to the  $s$ -support  $Sup_s(LT(C))$  to obtain a set  $\mathcal{U}' = Sup_s(LT(C)) \cup \{F\}$  of size  $|\mathcal{U}'| \leq s/2 + 1 \leq s$  such that  $R_{\mathcal{I}_E(Sup_s(LT(C)))}(C) = R_{\mathcal{I}_E(\mathcal{U}')}(C)$ . Since  $\mathcal{I}_E(\mathcal{U}')$  contains  $C$  as a generator we conclude that  $R_G(C) = R_{\mathcal{I}_E(\mathcal{U}')}(C) = 0$  also for  $C \in \mathcal{U}$ .

It remains to prove the last property in Lemma 3.14 stating that  $R_G(xt) = R_G(xR_G(t))$  for any term  $t$  such that  $Deg(t) < \lfloor (\delta s - 2\xi)/(2d) \rfloor$ . We can see that this holds by studying the following sequence of equalities:

$$\begin{aligned}
 R_G(xR_G(t)) &= \sum_{t' \in R_G(t)} R_G(xt') && \text{[by linearity]} \\
 &= \sum_{t' \in R_G(t)} R_{\mathcal{I}_E(Sup_s(xt'))}(xt') && \text{[by definition of } R_G\text{]} \\
 &= \sum_{t' \in R_G(t)} R_{\mathcal{I}_E(Sup_s(xt))}(xt') && \text{[by Lemma 3.26]} \\
 &= R_{\mathcal{I}_E(Sup_s(xt))}(xR_G(t)) && \text{[by linearity]} \\
 &= R_{\mathcal{I}_E(Sup_s(xt))}(xR_{\mathcal{I}_E(Sup_s(t))}(t)) && \text{[by definition of } R_G\text{]} \\
 &= R_{\mathcal{I}_E(Sup_s(xt))}(xt) && \text{[by Observation 3.12]} \\
 &= R_G(xt) && \text{[by definition of } R_G\text{]}
 \end{aligned}$$

Thus,  $R_G$  satisfies all the properties of Lemma 3.14, from which the theorem follows. ◀

We conclude the section by stating the following version of Theorem 3.7 for the most commonly occurring case with standard expansion without any slack.

► **Corollary 3.27.** *Suppose that  $(\mathcal{U}, \mathcal{V})_E$  is an  $(s, \delta, 0, E)$ -expander with overlap  $ol(\mathcal{V}) = d$  such that  $Vars(\mathcal{U} \wedge E) = \bigcup_{V \in \mathcal{V}} V$ . Then any polynomial calculus refutation of the formula  $\bigwedge_{F \in \mathcal{U}} F \wedge E$  requires degree strictly greater than  $\delta s/(2d)$ .*

**Proof sketch.** It is not hard to show that if a  $(\mathcal{U}, \mathcal{V})_E$ -graph is an  $(s, \delta, 0, E)$ -expander such that  $Vars(\mathcal{U} \wedge E) = \bigcup_{V \in \mathcal{V}} V$ , then for any  $\mathcal{U}' \subseteq \mathcal{U}$ ,  $|\mathcal{U}'| \leq s$ , it holds that the formula  $\mathcal{U}' \wedge E$  is satisfiable. Now the corollary follows immediately from Theorem 3.7. ◀

## 4 Applications

In this section, we demonstrate how to use the machinery developed in Section 3 to establish degree lower bounds for polynomial calculus. As a warm-up, let us consider the bound

<sup>6</sup> We remark that this is the only place in the proof where we are using that  $C$  is (the encoding of) a clause.

from [2] for CNF formulas  $\mathcal{F}$  whose clause-variable incidence graph  $G(\mathcal{F})$  are good enough expanders in the following sense.

► **Definition 4.1** (Bipartite boundary expander). A bipartite graph  $G = (U \dot{\cup} V, E)$  is a *bipartite  $(s, \delta)$ -boundary expander* if for every set of vertices  $U' \subseteq U$ ,  $|U'| \leq s$ , it holds that  $|\partial(U')| \geq \delta|U'|$ , where the *boundary*  $\partial(U') = \{v \in V : |N(v) \cap U'| = 1\}$  consists of all vertices on the right-hand side  $V$  that have a unique neighbour in  $U'$  on the left-hand side.

We can simply identify the  $(\mathcal{U}, \mathcal{V})_E$ -graph with the standard clause-variable incidence graph  $G(\mathcal{F})$  (setting  $E = \emptyset$ ) to recover the degree lower bound in [2] as stated next.

► **Theorem 4.2** ([2]). *For any CNF formula  $\mathcal{F}$  and any constant  $\delta > 0$  it holds that if the clause-variable incidence graph  $G(\mathcal{F})$  is an  $(s, \delta)$ -boundary expander, then the polynomial calculus degree required to refute  $\mathcal{F}$  in polynomial calculus is  $\text{Deg}(\mathcal{F} \vdash \perp) > \delta s/2$ .*

As a second application, which is more interesting in the sense that the  $(\mathcal{U}, \mathcal{V})_E$ -graph is nontrivial, we show how the degree lower bound for the ordering principle formulas in [12] can be established using this framework. For an undirected (and in general non-bipartite) graph  $G$ , the *graph ordering principle formula*  $GOP(G)$  says that there exists a totally ordered set of  $|V(G)|$  elements where no element is minimal, since every element/vertex  $v$  has a neighbour  $u \in N(v)$  which is smaller according to the ordering. Formally, the CNF formula  $GOP(G)$  is defined over variables  $x_{u,v}$ ,  $u, v \in V(G)$ ,  $u \neq v$ , where the intended meaning of the variables is that  $x_{u,v}$  is true if  $u < v$  according to the ordering, and consists of the following axiom clauses:

$$\bar{x}_{u,v} \vee \bar{x}_{v,w} \vee x_{u,w} \quad u, v, w \in V(G), u \neq v \neq w \neq u \quad (\text{transitivity}) \quad (4.1a)$$

$$\bar{x}_{u,v} \vee \bar{x}_{v,u} \quad u, v \in V(G), u \neq v \quad (\text{anti-symmetry}) \quad (4.1b)$$

$$x_{u,v} \vee x_{v,u} \quad u, v \in V(G), u \neq v \quad (\text{totality}) \quad (4.1c)$$

$$\bigvee_{u \in N(v)} x_{u,v} \quad v \in V(G) \quad (\text{non-minimality}) \quad (4.1d)$$

We remark that the graph ordering principle on the complete graph  $K_n$  on  $n$  vertices is the *(linear) ordering principle formula*  $LOP_n$  (also known as a *least number principle formula*, or *graph tautology* in the literature), for which the non-minimality axioms (4.1d) have width linear in  $n$ . By instead considering graph ordering formulas for graphs  $G$  of bounded degree, one can bring the initial width of the formulas down so that the question of degree lower bounds becomes meaningful.

To prove degree lower bounds for  $GOP(G)$  we need the following extension of boundary expansion to the case of non-bipartite graphs.

► **Definition 4.3** (Non-bipartite boundary expander). A graph  $G = (V, E)$  is an  *$(s, \delta)$ -boundary expander* if for every subset of vertices  $V' \subseteq V(G)$ ,  $|V'| \leq s$ , it holds that  $|\partial(V')| \geq \delta|V'|$ , where the *boundary*  $\partial(V') = \{v \in V(G) \setminus V' : |N(v) \cap V'| = 1\}$  is the set of all vertices in  $V(G) \setminus V'$  that have a unique neighbour in  $V'$ .

We want to point out that the definition of expansion used by Galesi and Lauria in [12] is slightly weaker in that they do not require boundary expansion but just vertex expansion (measured as  $|N(V') \setminus V'|$  for vertex sets  $V'$  with  $|V'| \leq s$ ), and hence their result is slightly stronger than what we state below in Theorem 4.4. With some modifications of the definition of  $E$ -respectful boundary in  $(\mathcal{U}, \mathcal{V})_E$ -graphs it would be possible to match the lower bound in [12], but it would also make the definitions more cumbersome and so we choose not to do so here.

► **Theorem 4.4** ([12]). *For a non-bipartite graph  $G$  that is an  $(s, \delta)$ -boundary expander it holds that  $\text{Deg}(GOP(G) \vdash \perp) > \delta s/4$ .*

**Proof sketch.** To form the  $(\mathcal{U}, \mathcal{V})_E$ -graph for  $GOP(G)$ , we let  $E$  consist of all transitivity axioms (4.1a), anti-symmetry axioms (4.1b), and totality axioms (4.1c). The non-minimality axioms (4.1d) viewed as singleton sets form the family  $\mathcal{U}$ , while  $\mathcal{V}$  is the family of variable sets  $V_v$  for each vertex  $v$  containing all variables that mention  $v$ , i.e.,  $V_v = \{x_{u,w} \mid u, w \in V(G), u = v \text{ or } w = v\}$ . We leave it to the reader to verify that  $(\mathcal{U}, \mathcal{V})_E$  is an  $(s, \delta, 0, E)$ -expander and that the overlap  $ol(\mathcal{V})$  is 2, which implies the lower bound. ◀

Let us now turn our attention back to bipartite graphs and consider different flavours of pigeonhole principle formulas. We will focus on formulas over bounded-degree bipartite graphs, where we will convert standard bipartite boundary expansion as in Definition 4.1 into respectful boundary expansion as in Definition 3.5. For a bipartite graph  $G = (U \dot{\cup} V, E)$  the axioms appearing in the different versions of the graph pigeonhole principle formulas are as follows:

$$\bigvee_{v \in N(u)} x_{u,v} \quad u \in U \quad \text{(pigeon axioms)} \quad (4.2a)$$

$$\bar{x}_{u,v} \vee \bar{x}_{u',v} \quad v \in V, u, u' \in N(v), u \neq u', \quad \text{(hole axioms)} \quad (4.2b)$$

$$\bar{x}_{u,v} \vee \bar{x}_{u,v'} \quad u \in U, v, v' \in N(u), v \neq v' \quad \text{(functionality axioms)} \quad (4.2c)$$

$$\bigvee_{u \in N(v)} x_{u,v} \quad v \in V \quad \text{(onto axioms)} \quad (4.2d)$$

The “plain vanilla” *graph pigeonhole principle formula*  $PHP_G$  is the CNF formula over variables  $\{x_{u,v} \mid (u, v) \in E\}$  consisting of clauses (4.2a) and (4.2b); the *graph functional pigeonhole principle formula*  $FPHP_G$  contains the clauses of  $PHP_G$  and in addition clauses (4.2c); the *graph onto pigeonhole principle formula*  $Onto-PHP_G$  contains  $PHP_G$  plus clauses (4.2d); and the *graph onto functional pigeonhole principle formula*  $Onto-FPHP_G$  consists of all the clauses (4.2a)–(4.2d).

We obtain the standard versions of the PHP formulas by considering graph formulas as above over the complete bipartite graph  $K_{n+1,n}$ . In the opposite direction, for any bipartite graph  $G$  with  $n + 1$  vertices on the left and  $n$  vertices on the right we can hit any version of the pigeonhole principle formula over  $K_{n+1,n}$  with the restriction  $\rho_G$  setting  $x_{u,v}$  to false for all  $(u, v) \notin E(G)$  to recover the corresponding graph pigeonhole principle formula over  $G$ . When doing so, we will use the observation from Section 2 that restricting a formula can only decrease the size and degree required to refute it.

As mentioned in Section 1, it was established already in [2] that good bipartite boundary expanders  $G$  yield formulas  $PHP_G$  that require large polynomial calculus degree to refute. We can reprove this result in our language – and, in fact, observe that the lower bound in [2] works also for the onto version  $Onto-PHP_G$  – by constructing an appropriate  $(\mathcal{U}, \mathcal{V})_E$ -graph. In addition, we can generalize the result in [2] slightly by allowing some additive slack  $\xi > 0$  in the expansion in Theorem 3.7. This works as long as we have the guarantee that no too small subformulas are unsatisfiable.

► **Theorem 4.5.** *Suppose that  $G = (U \dot{\cup} V, E)$  is a bipartite graph with  $|U| = n$  and  $|V| = n - 1$  and that  $\delta > 0$  is a constant such that*

- *for every set  $U' \subseteq U$  of size  $|U'| \leq s$  there is a matching of  $U'$  into  $V$ , and*
- *for every set  $U' \subseteq U$  of size  $|U'| \leq s$  it holds that  $|\partial(U')| \geq \delta|U'| - \xi$ .*

*Then  $\text{Deg}(Onto-PHP_G \vdash \perp) > \delta s/2 - \xi$ .*

**Proof sketch.** The  $(\mathcal{U}, \mathcal{V})_E$ -graph for  $PHP_G$  is formed by taking  $\mathcal{U}$  to be the set of pigeon axioms (4.2a),  $E$  to consist of the hole axioms (4.2b) and onto axioms (4.2d), and  $\mathcal{V}$  to be the collection of variable sets  $V_v = \{x_{u,v} \mid u \in N(v)\}$  partitioned with respect to the holes  $v \in V$ . It is straightforward to check that this  $(\mathcal{U}, \mathcal{V})_E$ -graph is isomorphic to the graph  $G$  and that all neighbours in  $(\mathcal{U}, \mathcal{V})_E$  are  $E$ -respectful (for  $\bigvee_{v \in N(u)} x_{u,v} \in \mathcal{U}$  and  $V_v$  for some  $v \in N(u)$ ), apply the partial assignment sending pigeon  $u$  to hole  $v$  and ruling out all other pigeons in  $N(v) \setminus \{u\}$  for  $v$ ). Moreover, using the existence of matchings for all sets of pigeons  $U'$  of size  $|U'| \leq s$  we can prove that every subformula  $\mathcal{U}' \wedge E$  is satisfiable as long as  $|\mathcal{U}'| \leq s$ . Hence, we can apply Theorem 3.7 to derive the claimed bound. We refer to the upcoming full-length version of [17] for the omitted details.  $\blacktriangleleft$

Theorem 4.5 is the only place in this paper where we use non-zero slack for the expansion. The reason that we need slack is so that we can establish lower bounds for another type of formulas, namely the subset cardinality formulas studied in [17, 28, 30]. A brief (and somewhat informal) description of these formulas is as follows. We start with a 4-regular bipartite graph to which we add an extra edge between two non-connected vertices. We then write down clauses stating that each degree-4 vertex on the left has at least 2 of its edges set to true, while the single degree-5 vertex has a strict majority of 3 incident edges set to true. On the right-hand side of the graph we encode the opposite, namely that all vertices with degree 4 have at least 2 of its edges set to false, while the vertex with degree 5 has at least 3 edges set to false. A simple counting argument yields that the CNF formula consisting of these clauses must be unsatisfiable. Formally, we have the following definition (which strictly speaking is a slightly specialized case of the general construction, but again we refer to [17] for the details).

► **Definition 4.6** (Subset cardinality formulas [17, 30]). Suppose that  $G = (U \dot{\cup} V, E)$  is a bipartite graph that is 4-regular except that one extra edge has been added between two unconnected vertices on the left and right. Then the *subset cardinality formula*  $SC(G)$  over  $G$  has variables  $x_e, e \in E$ , and clauses:

- $x_{e_1} \vee x_{e_2} \vee x_{e_3}$  for every triple  $e_1, e_2, e_3$  of edges incident to any  $u \in U$ ,
- $\bar{x}_{e_1} \vee \bar{x}_{e_2} \vee \bar{x}_{e_3}$  for every triple  $e_1, e_2, e_3$  of edges incident to any  $v \in V$ .

To prove lower bounds on refutation degree for these formulas we use the standard notion of vertex expansion on bipartite graphs, where all neighbours on the left are counted and not just unique neighbours as in Definition 4.1.

► **Definition 4.7** (Bipartite expander). A bipartite graph  $G = (U \dot{\cup} V, E)$  is a *bipartite*  $(s, \delta)$ -*expander* if for each vertex set  $U' \subseteq U, |U'| \leq s$ , it holds that  $|N(U')| \geq \delta|U'|$ .

The existence of such expanders with appropriate parameters can again be established by straightforward calculations (as in, for instance, [15]).

► **Theorem 4.8** ([17]). *Suppose that  $G = (U \dot{\cup} V, E)$  is a 4-regular bipartite  $(\gamma n, \frac{5}{2} + \delta)$ -expander for  $|U| = |V| = n$  and some constants  $\gamma, \delta > 0$ , and let  $G'$  be obtained from  $G$  by adding an arbitrary edge between two unconnected vertices in  $U$  and  $V$ . Then refuting the formula  $SC(G')$  requires degree  $\text{Deg}(SC(G') \vdash \perp) = \Omega(n)$ , and hence size  $S_{PCR}(SC(G') \vdash \perp) = \exp(\Omega(n))$ .*

**Proof sketch.** The proof is by reducing to graph PHP formulas and applying Theorem 4.5 (which of course also holds with onto axioms removed). We fix some complete matching in  $G$ , which is guaranteed to exist in regular bipartite graphs, and then set all edges in the

matching as well as the extra added edge to true. Now the degree-5 vertex  $v^*$  on the right has only 3 neighbours and the constraint for  $v^*$  requires all of these edges to be set to false. Hence, we set these edges to false as well which makes  $v^*$  and its clauses vanish from the formula. The restriction leaves us with  $n$  vertices on the left which require that at least 1 of the remaining 3 edges incident to them is true, while the  $n - 1$  vertices on the right require that at most 1 out of their incident edges is true. That is, we have restricted our subset cardinality formula to obtain a graph PHP formula.

As the original graph is a  $(\gamma n, \frac{5}{2} + \delta)$ -expander, a simple calculation can convince us that the new graph is a boundary expander where each set of vertices  $U'$  on the left with size  $|U'| \leq \gamma n$  has boundary expansion  $|\partial(U')| \geq 2\delta|U'| - 1$ . Note the additive slack of 1 compared to the usual expansion condition, which is caused by the removal of the degree-5 vertex  $v^*$  from the right. Now we can appeal to Theorem 4.5 (and Theorem 2.2) to obtain the lower bounds claimed in the theorem. ◀

Let us conclude this section by presenting our new lower bounds for the functional pigeonhole principle formulas. As a first attempt, we could try to reason as in the proof of Theorem 4.5 (but adding the axioms (4.2c) and removing axioms (4.2d)). The naive idea would be to modify our  $(\mathcal{U}, \mathcal{V})_E$ -graph slightly by substituting the functionality axioms for the onto axioms in  $E$  while keeping  $\mathcal{U}$  and  $\mathcal{V}$  the same. This does not work, however – although the sets  $V_v \in \mathcal{V}$  are  $E$ -respectful, the only assignment that respects  $E$  is the one that sets all variables  $x_{u,v} \in V_v$  to false. Thus, it is not possible to satisfy any of the pigeon axioms, meaning that there are no  $E$ -respectful neighbours in  $(\mathcal{U}, \mathcal{V})_E$ . In order to obtain a useful  $(\mathcal{U}, \mathcal{V})_E$ -graph, we instead need to redefine  $\mathcal{V}$  by enlarging the variable sets  $V_v$ , using the fact that  $\mathcal{V}$  is not required to be a partition. Doing so in the appropriate way yields the following theorem.

► **Theorem 4.9.** *Suppose that  $G = (U \dot{\cup} V, E)$  is a bipartite  $(s, \delta)$ -boundary expander with left degree bounded by  $d$ . Then it holds that refuting  $FPHP_G$  in polynomial calculus requires degree strictly greater than  $\delta s / (2d)$ . It follows that if  $G$  is a bipartite  $(\gamma n, \delta)$ -boundary expander with constant left degree and  $\gamma, \delta > 0$ , then any polynomial calculus (PC or PCR) refutation of  $FPHP_G$  requires size  $\exp(\Omega(n))$ .*

**Proof.** We construct a  $(\mathcal{U}, \mathcal{V})_E$ -graph from  $FPHP_G$  as follows. We let the set of clauses  $E$  consist of all hole axioms (4.2b) and functionality axioms (4.2c). We define the family  $\mathcal{U}$  to consist of the pigeon axioms (4.2a) interpreted as singleton CNF formulas. For the variables we let  $\mathcal{V} = \{V_v \mid v \in V\}$ , where for every hole  $v \in V$  the set  $V_v$  is defined by

$$V_v = \{x_{u',v'} \mid u' \in N(v) \text{ and } v' \in N(u')\} . \tag{4.3}$$

That is, to build  $V_v$  we start with the hole  $v$  on the right, consider all pigeons  $u'$  on the left that can go into this hole, and finally include in  $V_v$  for all such  $u'$  the variables  $x_{u',v'}$  for all holes  $v'$  incident to  $u'$ . We want to show that  $(\mathcal{U}, \mathcal{V})_E$  as defined above satisfies the conditions in Corollary 3.27.

Note first that every variable set  $V_v$  respects the clause set  $E$  since setting all variables in  $V_v$  to false satisfies all clauses in  $E$  mentioning variables in  $V_v$ . It is easy to see from (4.3) that when a hole  $v$  is a neighbour of a pigeon  $u$ , the variable set  $V_v$  is also a neighbour in the  $(\mathcal{U}, \mathcal{V})_E$ -graph of the corresponding pigeon axiom  $F_u = \bigvee_{v \in N(u)} x_{u,v}$ . These are the only neighbours of the pigeon axiom  $F_u$ , as each  $V_v$  contains only variables mentioning pigeons in the neighbourhood of  $v$ . In other words,  $G$  and  $(\mathcal{U}, \mathcal{V})_E$  share the same neighbourhood structure.



Moreover, we claim that every neighbour  $V_v$  of  $F_u$  is an  $E$ -respectful neighbour. To see this, consider the assignment  $\rho_{u,v}$  that sets  $x_{u,v}$  to true and the remaining variables in  $V_v$  to false. Clearly,  $F_u$  is satisfied by  $\rho_{u,v}$ . All axioms in  $E$  not containing  $x_{u,v}$  are either satisfied by  $\rho_{u,v}$  or left untouched, since  $\rho_{u,v}$  assigns all other variables in its domain to false. Any hole axiom  $\bar{x}_{u,v} \vee \bar{x}_{u',v}$  in  $E$  that *does* contain  $x_{u,v}$  is satisfied by  $\rho_{u,v}$  since  $x_{u',v} \in V_v$  for  $u' \in N(v)$  by (4.3) and this variable is set to false by  $\rho_{u,v}$ . In the same way, any functionality axiom  $\bar{x}_{u,v} \vee \bar{x}_{u,v'}$  containing  $x_{u,v}$  is satisfied since the variable  $x_{u,v'}$  is in  $V_v$  by (4.3) and is hence assigned to false. Thus, the assignment  $\rho_{u,v}$   $E$ -respectfully satisfies  $F_u$ , and so  $F_u$  and  $V_v$  are  $E$ -respectful neighbours as claimed.

Since our constructed  $(\mathcal{U}, \mathcal{V})_E$ -graph is isomorphic to the original graph  $G$  and all neighbour relations are respectful, the expansion parameters of  $G$  trivially carry over to respectful expansion in  $(\mathcal{U}, \mathcal{V})_E$ . This is just another way of saying that  $(\mathcal{U}, \mathcal{V})_E$  is an  $(s, \delta, 0, E)$ -expander.

To finish the proof, note that the overlap of  $\mathcal{V}$  is at most  $d$ . This is so since a variable  $x_{u,v}$  appears in a set  $V_{v'}$  only when  $v' \in N(u)$ . Hence, for all variables  $x_{u,v}$  it holds that they appear in at most  $|N(u)| \leq d$  sets in  $\mathcal{V}$ . Now the conclusion that any polynomial calculus refutation of  $FPHP_G$  requires degree greater than  $\delta s / (2d)$  can be read off from Corollary 3.27. In addition, the exponential lower bound on the size of a refutation of  $FPHP_G$  when  $G$  is a  $(\gamma n, \delta)$ -boundary expander  $G$  with constant left degree follows by plugging the degree lower bound into Theorem 2.2.  $\blacktriangleleft$

It is not hard to show (again we refer to [15] for the details) that there exist bipartite graphs with left degree 3 which are  $(\gamma n, \delta)$ -boundary expanders for  $\gamma, \delta > 0$  and hence our size lower bound for polynomial calculus refutations of  $FPHP_G$  can be applied to them. Moreover, if  $|U| = n + 1$  and  $|V| = n$ , then we can identify some bipartite graph  $G$  that is a good expander and hit  $FPHP_n^{n+1} = FPHP_{K_{n+1,n}}$  with a restriction  $\rho_G$  setting  $x_{u,v}$  to false for all  $(u, v) \notin E$  to obtain  $FPHP_n^{n+1} \upharpoonright_{\rho_G} = FPHP_G$ . Since restrictions can only decrease refutation size, it follows that size lower bounds for  $FPHP_G$  apply also to  $FPHP_n^{n+1}$ , yielding the second lower bound claimed in Section 1.1.

► **Theorem 4.10.** *Any polynomial calculus or polynomial calculus resolution refutation of (the standard CNF encoding of) the functional pigeonhole principle  $FPHP_n^{n+1}$  requires size  $\exp(\Omega(n))$ .*

## 5 Concluding Remarks

In this work, we extend the techniques developed by Alekhovich and Razborov [2] for proving degree lower bounds on refutations of CNF formulas in polynomial calculus. Instead of looking at the clause-variable incidence graph  $G(F)$  of the formula  $F$  as in [2], we allow clustering of clauses and variables and reason in terms of the incidence graph  $G'$  defined on these clusters. We show that the CNF formula  $F$  requires high degree to be refuted in polynomial calculus whenever this clustering can be done in a way that “respects the structure” of the formula and so that the resulting graph  $G'$  has certain expansion properties.

This provides us with a unified framework within which we can reprove previously established degree lower bounds in [2, 12, 17]. More importantly, this also allows us to obtain a degree lower bound on the functional pigeonhole principle defined on expander graphs, solving an open problem from [23]. It immediately follows from this that the (standard CNF encodings of) the usual functional pigeonhole principle formulas require exponential proof size in polynomial calculus resolution, resolving a question on Razborov’s problems list [26]



which had (quite annoyingly) remained open. This means that we now have an essentially complete understanding of how the different variants of pigeonhole principle formulas behave with respect to polynomial calculus in the standard setting with  $n + 1$  pigeons and  $n$  holes. Namely, while Onto-FPHP formulas are easy, both FPHP formulas and Onto-PHP formulas are exponentially hard in  $n$  even when restricted to bounded-degree expanders.

A natural next step would be to see if this generalized framework can also be used to attack other interesting formula families which are known to be hard for resolution but for which there are currently no lower bounds in polynomial calculus. In particular, can our framework or some modification of it prove a lower bound for refuting the formulas encoding that a graph does not contain an independent set of size  $k$ , which were proven hard for resolution in [4]? Or what about the formulas stating that a graph is  $k$ -colorable, for which resolution lower bounds were established in [3]?

Returning to the pigeonhole principle, we now understand how different encodings behave in polynomial calculus when we have  $n + 1$  pigeons and  $n$  holes. But what happens when we increase the number of pigeons? For instance, do the formulas become easier if we have  $n^2$  pigeons and  $n$  holes? (This is the point where lower bound techniques based on degree break down.) What about arbitrary many pigeons? In resolution these questions are fairly well understood, as witnessed by the works of Raz [20] and Razborov [22, 24, 25], but as far as we are aware they remain wide open for polynomial calculus.

**Acknowledgements.** We are grateful to Ilario Bonacina, Yuval Filmus, Nicola Galesi, Alexander Razborov, and Marc Vinyals for numerous discussions on proof complexity in general and polynomial calculus degree lower bounds in particular. We want to give a special thanks to Massimo Lauria for several insightful comments on a preliminary version of this work, which helped to simplify the construction (and improve the parameters in the results) considerably. Finally, we are thankful for the helpful comments from the anonymous referees.

The authors were funded by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007–2013) / ERC grant agreement no. 279611. The second author was also supported by Swedish Research Council grants 621-2010-4797 and 621-2012-5645.

---

## 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 appeared 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 appeared in *FOCS'01*.
- 3 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.
- 4 Paul Beame, Russell Impagliazzo, and Ashish Sabharwal. The resolution complexity of independent sets and vertex covers in random graphs. *Computational Complexity*, 16(3):245–297, October 2007.
- 5 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 appeared in *STOC'99*.

- 6 Archie Blake. *Canonical Expressions in Boolean Algebra*. PhD thesis, University of Chicago, 1937.
- 7 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 appeared in *CCC'99*.
- 8 Vašek Chvátal and Endre Szemerédi. Many hard examples for resolution. *Journal of the ACM*, 35(4):759–768, October 1988.
- 9 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.
- 10 Stephen A. Cook and Robert Reckhow. The relative efficiency of propositional proof systems. *Journal of Symbolic Logic*, 44(1):36–50, March 1979.
- 11 Yuval Filmus. On the Alekhovich–Razborov degree lower bound for the polynomial calculus. Manuscript. Available at <http://www.cs.toronto.edu/~yuvalf/A1Ra.pdf>, 2014.
- 12 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.
- 13 Dima Grigoriev. Tseitin's tautologies and lower bounds for Nullstellensatz proofs. In *Proceedings of the 39th Annual IEEE Symposium on Foundations of Computer Science (FOCS'98)*, pages 648–652, November 1998.
- 14 Armin Haken. The intractability of resolution. *Theoretical Computer Science*, 39(2-3):297–308, August 1985.
- 15 Shlomo Hoory, Nathan Linial, and Avi Wigderson. Expander graphs and their applications. *Bulletin of the American Mathematical Society*, 43(4):439–561, October 2006.
- 16 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.
- 17 Mladen Mikša and Jakob Nordström. Long proofs of (seemingly) simple formulas. In *Proceedings of the 17th International Conference on Theory and Applications of Satisfiability Testing (SAT'14)*, volume 8561 of *Lecture Notes in Computer Science*, pages 121–137. Springer, July 2014.
- 18 Mladen Mikša and Jakob Nordström. A generalized method for proving polynomial calculus degree lower bounds. Technical Report TR15-078, Electronic Colloquium on Computational Complexity (ECCC), May 2015.
- 19 Jakob Nordström. Pebble games, proof complexity and time-space trade-offs. *Logical Methods in Computer Science*, 9:15:1–15:63, September 2013.
- 20 Ran Raz. Resolution lower bounds for the weak pigeonhole principle. *Journal of the ACM*, 51(2):115–138, March 2004. Preliminary version appeared in *STOC'02*.
- 21 Alexander A. Razborov. Lower bounds for the polynomial calculus. *Computational Complexity*, 7(4):291–324, December 1998.
- 22 Alexander A. Razborov. Improved resolution lower bounds for the weak pigeonhole principle. Technical Report TR01-055, Electronic Colloquium on Computational Complexity (ECCC), July 2001.
- 23 Alexander A. Razborov. Proof complexity of pigeonhole principles. In *5th International Conference on Developments in Language Theory, (DLT'01), Revised Papers*, volume 2295 of *Lecture Notes in Computer Science*, pages 100–116. Springer, July 2002.
- 24 Alexander A. Razborov. Resolution lower bounds for the weak functional pigeonhole principle. *Theoretical Computer Science*, 1(303):233–243, June 2003.
- 25 Alexander A. Razborov. Resolution lower bounds for perfect matching principles. *Journal of Computer and System Sciences*, 69(1):3–27, August 2004. Preliminary version appeared in *CCC'02*.

- 26 Alexander A. Razborov. Possible research directions. List of open problems (in proof complexity and other areas) available at <http://people.cs.uchicago.edu/~razborov/teaching/>, 2014.
- 27 Søren Riis. *Independence in Bounded Arithmetic*. PhD thesis, University of Oxford, 1993.
- 28 Ivor Spence. sgen1: A generator of small but difficult satisfiability benchmarks. *Journal of Experimental Algorithmics*, 15:1.2:1.1–1.2:1.15, March 2010.
- 29 Alasdair Urquhart. Hard examples for resolution. *Journal of the ACM*, 34(1):209–219, January 1987.
- 30 Allen Van Gelder and Ivor Spence. Zero-one designs produce small hard SAT instances. In *Proceedings of the 13th International Conference on Theory and Applications of Satisfiability Testing (SAT'10)*, volume 6175 of *Lecture Notes in Computer Science*, pages 388–397. Springer, July 2010.

# Generalized Quantum Arthur-Merlin Games

Hirotsada Kobayashi<sup>1</sup>, François Le Gall<sup>2</sup>, and Harumichi Nishimura<sup>3</sup>

- 1 Principles of Informatics Research Division  
National Institute of Informatics  
2-1-2 Hitotsubashi, Chiyoda, Tokyo 101-8430, Japan  
hirotada@nii.ac.jp
- 2 Department of Computer Science  
Graduate School of Information Science and Technology  
The University of Tokyo  
7-3-1 Hongo, Bunkyo, Tokyo 113-0033, Japan  
legall@is.s.u-tokyo.ac.jp
- 3 Department of Computer Science and Mathematical Informatics  
Graduate School of Information Science  
Nagoya University  
Furo-cho, Chikusa, Nagoya, Aichi 464-8601, Japan  
hnishimura@is.nagoya-u.ac.jp

---

## Abstract

This paper investigates the role of interaction and coins in *quantum Arthur-Merlin games* (also called *public-coin quantum interactive proof systems*). While the existing model restricts the messages from the verifier to be classical even in the quantum setting, the present work introduces a generalized version of quantum Arthur-Merlin games where the messages from the verifier can be quantum as well: the verifier can send not only random bits, but also halves of EPR pairs. This generalization turns out to provide several novel characterizations of quantum interactive proof systems with a constant number of turns. First, it is proved that the complexity class corresponding to two-turn quantum Arthur-Merlin games where both of the two messages are quantum, denoted qq-QAM in this paper, does not change by adding a constant number of turns of classical interaction prior to the communications of qq-QAM proof systems. This can be viewed as a quantum analogue of the celebrated collapse theorem for AM due to Babai. To prove this collapse theorem, this paper presents a natural complete problem for qq-QAM: deciding whether the output of a given quantum circuit is close to a totally mixed state. This complete problem is on the very line of the previous studies investigating the hardness of checking properties related to quantum circuits, and thus, qq-QAM may provide a good measure in computational complexity theory. It is further proved that the class qq-QAM<sub>1</sub>, the perfect-completeness variant of qq-QAM, gives new bounds for standard well-studied classes of two-turn quantum interactive proof systems. Finally, the collapse theorem above is extended to comprehensively classify the role of classical and quantum interactions in quantum Arthur-Merlin games: it is proved that, for any constant  $m \geq 2$ , the class of problems having  $m$ -turn quantum Arthur-Merlin proof systems is either equal to PSPACE or equal to the class of problems having two-turn quantum Arthur-Merlin proof systems of a specific type, which provides a complete set of quantum analogues of Babai's collapse theorem.

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

**Keywords and phrases** interactive proof systems, Arthur-Merlin games, quantum computing, complete problems, entanglement

**Digital Object Identifier** 10.4230/LIPIcs.CCC.2015.488



© Hirotsada Kobayashi, François Le Gall, and Harumichi Nishimura;  
licensed under Creative Commons License CC-BY

30th Conference on Computational Complexity (CCC'15).

Editor: David Zuckerman; pp. 488–511



Leibniz International Proceedings in Informatics

LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

## 1 Introduction

**Background and motivation.** Interactive proof systems [9, 4] play a central role in computational complexity and have many applications such as probabilistically checkable proofs and zero-knowledge proofs. The aim of such a system is the verification of an assertion (e.g., verifying if an input is in a language) by a party implementing a polynomial-time probabilistic computation, called the verifier, interacting with another party with unlimited power, called the prover, in polynomially many turns. Two definitions are given on the secrecy of the coin which the verifier can flip: Goldwasser, Micali, and Rackoff [9] defined private-coin proof systems, where the prover cannot see the outcomes of coin flips, while Babai [4] defined public-coin proof systems, where the prover can see all the outcomes of coin flips. Public-coin interactive proof systems are often called Arthur-Merlin games or Arthur-Merlin proof systems, since the verifier was called Arthur and the prover was called Merlin in Ref. [4].

It is natural to expect that the power of interactive proof systems depends on the number of turns of interaction. Babai [4] showed, however, that as long as the number of turns is a constant at least two, the number of turns does not affect the power of Arthur-Merlin proof systems, i.e.,  $AM(m) = AM(2)$  for any constant  $m \geq 2$  (the *collapse theorem*), where  $AM(m)$  is the class of problems having  $m$ -turn Arthur-Merlin proof systems. Goldwasser and Sipser [10] then showed that a private-coin interactive proof system can be simulated by an Arthur-Merlin proof system by adding two turns, and thus, these two types of interactive proof systems are computationally equivalent. By the above results, the class of problems having interactive proof systems of a constant number of turns is equal to  $AM(2)$  (regardless of definitions with public coins or private coins), and this class is nowadays called  $AM$ . The class  $AM$  is believed to be much smaller than  $PSPACE$ , as it is contained in  $\Pi_2^P$  in the second level of the polynomial-time hierarchy [22, 4]. On the contrary, the class of problems having more general interactive proof systems of polynomially many turns, called  $IP$ , does coincide with  $PSPACE$  [26, 23, 28] (again regardless of definitions with public coins or private coins [10, 29]).

Quantum interactive proof systems were introduced by Watrous [34], and the class of problems having quantum interactive proof systems is called  $QIP$ . In the quantum world, the importance of the number of turns in interactive proof systems is drastically changed. The first paper on quantum interactive proofs [34] already proved the surprising power of quantum interactive proof systems with a constant number of turns, by showing that any problem in  $PSPACE$  has a three-turn quantum interactive proof system. Kitaev and Watrous [16] then proved that any quantum interactive proof system can be simulated by a three-turn quantum interactive proof system, namely,  $QIP = QIP(3)$ , where  $QIP(m)$  denotes the class of problems having  $m$ -turn quantum interactive proof systems. Finally, the recent result  $QIP = PSPACE$  by Jain, Ji, Upadhyay, and Watrous [13] completely characterized the computational power of quantum interactive proof systems with three turns or more. In contrast, despite a number of intensive studies [33, 36, 14, 12], still very little is known on the class  $QIP(2)$  corresponding to *two-turn* quantum interactive proof systems, and characterizing the computational power of two-turn quantum interactive proof systems is one of the main open problems in this field.

A public-coin version of quantum interactive proof systems was first introduced by Marriott and Watrous [24], named quantum Arthur-Merlin proof systems, where the messages from the verifier are restricted to classical strings consisting only of outcomes of polynomially many attempts of a fair coin flip. They then showed that three-turn quantum Arthur-Merlin proof systems can simulate three-turn standard quantum interactive proof systems, and

hence the corresponding class, denoted QMAM, coincides with  $\text{QIP} = \text{PSPACE}$ . They also investigated the case of two-turn quantum Arthur-Merlin proof systems and showed that the corresponding class, denoted QAM, is included in  $\text{BP} \cdot \text{PP}$ , a subclass of PSPACE obtained by applying the BP operator to the class PP, which is still the only nontrivial upper bound known for QAM.

**Results and their meanings.** This paper introduces a “fully quantum” version of quantum Arthur-Merlin proof systems, which generalizes the existing quantum Arthur-Merlin proof systems in Ref. [24]. In this generalized model, the verifier can send quantum messages, but these messages can be used only for sharing EPR pairs with the prover, i.e., the verifier at his/her turn first generates polynomially many EPR pairs and then sends one half of each of them to the prover. Recall that classical public-coin messages can be interpreted as messages for sharing uniform randomness between the verifier and the prover. In this context, sharing EPR pairs would be the most natural quantum analogue of sharing randomness, and thus, the model introduced above may be viewed as a natural full-quantum version of quantum Arthur-Merlin proof systems.

The main interest in this model is again on the two-turn case, as allowing three or more turns in this model obviously hits the PSPACE ceiling. Let qq-QAM be the class of problems having two-turn “fully quantum” Arthur-Merlin proof systems, i.e., two-turn quantum interactive proof systems in which the first message from the verifier consists only of polynomially many halves of EPR pairs. Note that the only difference from the existing class QAM lies in the type of the message from the verifier: uniform random classical bits are replaced by halves of EPR pairs. The main goal of this paper is to investigate the computational power of this class qq-QAM in order to figure out the advantages offered by sharing EPR pairs rather than classical randomness, and more generally, to make a step forward in the understanding of two-turn quantum interactive proof systems.

While the class qq-QAM is the main target of investigation, this paper further studies the power of various models of quantum Arthur-Merlin proofs with quantum/classical messages. For any constant  $m \geq 1$  and any message-types  $t_1, \dots, t_m$  in  $\{c, q\}$ , let  $t_m \cdots t_1$ -QAM( $m$ ) be the class of problems having  $m$ -turn quantum interactive proof systems with the following restrictions:

- For any odd  $j$ ,  $1 \leq j \leq m$ , the  $(m - j + 1)$ st message (or the  $j$ th message counting from the last), which is the message from the prover sent at the  $(m - j + 1)$ st turn, is a quantum message if  $t_j = q$ , and is restricted to a classical message if  $t_j = c$ .
- For any even  $j$ ,  $1 \leq j \leq m$ , at the  $(m - j + 1)$ st turn, which is a turn for the verifier, the verifier first generates polynomially many EPR pairs and then sends halves of them if  $t_j = q$ , while the verifier first flips a fair coin polynomially many times and then sends their outcomes if  $t_j = c$ .

The class  $t_m \cdots t_1$ -QAM( $m$ ) may be simply written as  $t_m \cdots t_1$ -QAM when there is no ambiguity in the number of turns: for instance, qq-QAM(2) may be abbreviated to qq-QAM. Note that the classes QAM and QMAM defined in Ref. [24] are exactly the classes cq-QAM and qcq-QAM, respectively. The class cc-QAM corresponds to two-turn public-coin quantum interactive proofs with classical communications: the verifier sends a question consisting only of outcomes of polynomially many attempts of a fair coin flip, then the prover responds with polynomially many classical bits, and the final verification is done by the verifier via polynomial-time quantum computation. By definition,  $\text{AM} \subseteq \text{cc-QAM} \subseteq \text{cq-QAM} \subseteq \text{qq-QAM} \subseteq \text{QIP}(2)$ .

As mentioned above, the main target in this paper is the class qq-QAM. First, it is proved that the power of qq-QAM proof systems does not change by adding a constant number of turns of classical interaction prior to the communications of qq-QAM proof systems.



► **Theorem 1.1.** *For any constant  $m \geq 2$ ,  $c \cdots c$ qQ-QAM( $m$ ) = qq-QAM.*

In stark contrast to this, as mentioned before and will be stated clearly in Theorem 1.7, adding one turn of prior quantum interaction gives qq-QAM proof systems the full power of quantum interactive proof systems (i.e., the resulting class is PSPACE). Hence, Theorem 1.1 may be viewed as a quantum analogue of Babai's collapse theorem [4] for the class qq-QAM.

The proof of Theorem 1.1 comes in three parts: The first part proves that, for any constant  $m \geq 4$ ,  $c \cdots c$ qQ-QAM( $m$ ) is necessarily included in ccqQ-QAM. The second part proves that ccqQ-QAM is included in qq-QAM. Finally, the third part proves that ccqQ-QAM is included in qq-QAM, by using the containment proved in the second part.

The first part is proved by carefully extending the argument in Babai's collapse theorem. The core idea of Babai's proof is that, by a probabilistic argument applied to a parallel repetition of the original proof system, the order of the verifier and the prover in the first three turns of the original system can be switched, which results in another proof system that has fewer number of turns. When proving the first part, the messages of the first three turns of the original  $m$ -turn quantum Arthur-Merlin proof system are classical, and thus, the argument in Babai's collapse theorem still works.

The proof of the second part is one of the highlights of this paper. The main difficulty in proving this part (and the third part) is that the argument used in Babai's collapse theorem fails when any of the first three turns is quantum in the starting proof system.

To overcome this difficulty, this paper first provides a natural complete promise problem for qq-QAM, namely, the CLOSE IMAGE TO TOTALLY MIXED (CITM) problem, which asks to check if the image of a given quantum circuit can be close to a totally mixed state, formally defined as follows.

---

CLOSE IMAGE TO TOTALLY MIXED PROBLEM: CITM( $a, b$ )

**Input:** A description of a quantum circuit  $Q$  acting on  $q_{\text{all}}$  qubits that has  $q_{\text{in}}$  specified input qubits and  $q_{\text{out}}$  specified output qubits.

**Yes Instances:** There exists a  $q_{\text{in}}$ -qubit state  $\rho$  such that  $D(Q(\rho), (I/2)^{\otimes q_{\text{out}}}) \leq a$ .

**No Instances:** For any  $q_{\text{in}}$ -qubit state  $\rho$ ,  $D(Q(\rho), (I/2)^{\otimes q_{\text{out}}}) \geq b$ .

---

Here,  $D(\cdot, \cdot)$  denotes the trace distance,  $Q(\rho)$  is the  $q_{\text{out}}$ -qubit output state of  $Q$  when the input state was  $\rho$  (i.e., the reduced state obtained by tracing out the space corresponding to the  $(q_{\text{all}} - q_{\text{out}})$  non-output qubits after applying  $Q$  to  $\rho \otimes (|0\rangle\langle 0|)^{\otimes (q_{\text{all}} - q_{\text{in}})}$ ), and  $I$  is the identity operator of dimension two (and thus,  $(I/2)^{\otimes q_{\text{out}}}$  corresponds to the totally mixed state of  $q_{\text{out}}$  qubits). The following completeness result is proved.

► **Theorem 1.2.** *For any constants  $a$  and  $b$  in  $(0, 1)$  such that  $(1 - a)^2 > 1 - b^2$ , CITM( $a, b$ ) is qq-QAM-complete under polynomial-time many-one reduction.*

Then the core idea for proving the second part is to use the structure of this complete problem that yes-instances are witnessed by the existence of a quantum state (i.e., the  $\exists$  quantifier appears in the first place), while no such witness quantum state exists for no-instances (i.e., the  $\forall$  quantifier appears in the first place). This makes it possible to incorporate the first turn of the ccqQ-QAM system into the input quantum state of the complete problem CITM (as the quantifier derived from the first turn of the ccqQ-QAM system matches the quantifier derived from the complete problem CITM), and thus, any problem in ccqQ-QAM



can be reduced in polynomial time to the CITM problem with appropriate parameters, which is in qq-QAM.

Actually, for the proof, whether the image of a constructed quantum circuit can be close to a totally mixed state is partly evaluated by using the maximum output entropy of quantum channels, which shows implicitly the qq-QAM-completeness of another problem that asks to check whether the maximum output entropy of a quantum channel is larger than a given value or not. More formally, the following MAXIMUM OUTPUT QUANTUM ENTROPY APPROXIMATION (MAXOUTQEA) problem is also qq-QAM-complete.

---

MAXIMUM OUTPUT QUANTUM ENTROPY APPROXIMATION PROBLEM: MAXOUTQEA

**Input:** A description of a quantum circuit that specifies a quantum channel  $\Phi$ , and a positive integer  $t$ .

**Yes Instances:**  $S_{\max}(\Phi) \geq t + 1$ .

**No Instances:**  $S_{\max}(\Phi) \leq t - 1$ .

---

Here,  $S_{\max}(\cdot)$  denotes the maximum output von Neumann entropy. Namely, for any quantum channel  $\Phi$ ,  $S_{\max}(\Phi) = \max_{\rho} S(\Phi(\rho))$ , where  $S(\cdot)$  denotes the von Neumann entropy and  $\Phi(\rho)$  is the output quantum state of  $\Phi$  when the input quantum state to it was  $\rho$ .

► **Theorem 1.3.** MAXOUTQEA is qq-QAM-complete under polynomial-time many-one reduction.

Finally, the third part of the proof of Theorem 1.1 is obtained by first providing a randomized reduction from a problem in ccq-QAM to a problem in cq-QAM, and then using the containment proved in the second part for the resulting problem in cq-QAM.

Besides its usefulness in proving Theorem 1.1, the complete problem CITM is of independent interest in the following sense. Recall that problems with formulations similar to CITM have already been studied, and were crucial to understand and characterize several complexity classes related to quantum interactive proof systems: testing closeness between the images of two given quantum circuits is QIP-complete [27] (and hence PSPACE-complete), testing closeness between the state produced by a given circuit and the image of another quantum circuit is QIP(2)-complete [32] (see also Ref. [12]), testing closeness between the two states produced by two given quantum circuits is QSZK-complete [33, 35], and testing closeness between the state produced by a quantum circuit and the totally mixed state is NISZK-complete [18, 8]. Theorem 1.2 shows that the class qq-QAM, besides its theoretical interest in the context of interactive proofs, is a very natural one that actually corresponds to a concrete computational problem that is on this line of studies investigating the hardness of checking properties related to quantum circuits. Since CITM corresponds to the remaining pattern (image versus totally mixed state), Theorem 1.2 provides the last piece for characterizing the hardness of these kinds of computational problems.

The complete problem MAXOUTQEA is also on the very line of the previous studies. Indeed, it is known that the following problems characterize the power of various models of quantum interactive proofs: deciding which of the two states produced by two given quantum circuits has higher entropy is QSZK-complete [6], and checking whether the entropy of the state produced by a given quantum circuit is larger than a given value or not is NISZK-complete [6, 8]. Along this line, MAXOUTQEA is the first entropy-related problem

that characterizes the power of quantum interactive proofs without zero-knowledge property, which may be worthy of note.

It is further proved that the class cq-QAM (i.e., the standard QAM) is necessarily contained in the one-sided bounded error version of qq-QAM of perfect completeness, denoted by qq-QAM<sub>1</sub> (throughout this paper, the perfect completeness version of each complexity class is indicated by adding the subscript “1”).

► **Theorem 1.4.**  $\text{cq-QAM} \subseteq \text{qq-QAM}_1$ .

One useful property when proving this theorem is that the proof of Theorem 1.1 does not harm the perfect completeness property, i.e., the equality  $c \cdots \text{cq-QAM}_1(m) = \text{qq-QAM}_1$  also holds for any constant  $m \geq 2$ . Especially, the class ccq-QAM<sub>1</sub> is included in the class qq-QAM<sub>1</sub>, and thus, one has only to prove that cq-QAM is included in ccq-QAM<sub>1</sub>. This can be proved by combining the classical technique due to Cai [7] for proving  $\text{AM} = \text{AM}_1$  (which itself originates in the proof of  $\text{BPP} \subseteq \Sigma_2^P$  due to Lautemann [22]), and the recent result that any problem in QMA has a one-sided bounded error quantum Merlin-Arthur proof system of perfect completeness in which Arthur and Merlin initially share a constant number of EPR pairs [20] (which in particular implies that QMA is included in qq-QAM<sub>1</sub>). Now the point is that, using two classical turns, the classical technique in Ref. [7] can be used to generate polynomially many instances of a (promise) QMA problem, all of which are QMA yes-instances if the input was a yes-instance, while at least one of which is a QMA no-instance with high probability if the input was a no-instance. Hence, by making use of the proof system in Ref. [20] for each QMA instance, which essentially runs polynomially many attempts of a protocol of qq-QAM type in parallel to check that none of them results in rejection, one obtains a proof system of ccq-QAM type with perfect completeness.

An immediate corollary of this theorem is the first nontrivial upper bound for QAM in terms of quantum interactive proofs.

► **Corollary 1.5.**  $\text{QAM} \subseteq \text{QIP}_1(2)$ .

Here,  $\text{QIP}_1(2)$  denotes the class of problems having two-turn quantum interactive proof systems of perfect completeness. This also improves the best known lower bound of  $\text{QIP}_1(2)$  (from QMA shown in Ref. [20] to QAM). By using the fact  $\text{MQA} = \text{MQA}_1$  (a.k.a.,  $\text{QCMA} = \text{QCMA}_1$ ) stating that classical-witness QMA systems can be made perfectly complete [15], a technique similar to the proof of Theorem 1.4 proves that perfect completeness is achievable in cc-QAM.

► **Theorem 1.6.**  $\text{cc-QAM} = \text{cc-QAM}_1$ .

Finally, results similar to Theorem 1.1 can be derived for other complexity classes related to generalized quantum Arthur-Merlin proof systems. Namely, the following complete characterization is proved on the power of generalized quantum Arthur-Merlin proofs involving a constant number of turns, which can be viewed as the complete set of quantum analogues of Babai’s collapse theorem.

► **Theorem 1.7.** *The following four properties hold:*

- (i) *For any constant  $m \geq 3$  and any message-types  $t_1, \dots, t_m$  in  $\{c, q\}$ , if there exists an index  $j \geq 3$  such that  $t_j = q$ , then  $t_m \cdots t_1\text{-QAM}(m) = \text{PSPACE}$ .*
- (ii) *For any constant  $m \geq 2$  and any message-type  $t$  in  $\{c, q\}$ ,  $c \cdots \text{cqt-QAM}(m) = \text{qq-QAM}$ .*
- (iii) *For any constant  $m \geq 2$ ,  $c \cdots \text{cq-QAM}(m) = \text{cq-QAM}$  (= QAM).*
- (iv) *For any constant  $m \geq 2$ ,  $c \cdots \text{c-QAM}(m) = \text{cc-QAM}$ .*

**Further related work.** There are several studies in which relevant subclasses of qq-QAM were treated. The class  $\text{QMA}^{\text{const-EPR}}$  was introduced in Ref. [20] to give an upper bound of QMA by its one-sided bounded error subclass  $\text{QMA}_1^{\text{const-EPR}}$  with perfect completeness. This  $\text{QMA}^{\text{const-EPR}}$  is an obvious subclass of qq-QAM with a restriction that the first message from the verifier consists of not polynomially many but a constant number of halves of EPR pairs. The class qq-QAM may be called  $\text{QMA}^{\text{poly-EPR}}$ , following the notation in Ref. [20]. Another subclass of qq-QAM is the class NIQSZK studied in Refs. [18, 8] that corresponds to non-interactive quantum statistical zero-knowledge proof systems, where the zero-knowledge property must also be satisfied.

**Organization of the paper.** Section 2 summarizes the notions and properties that are used throughout this paper. Section 3 presents formal definitions of generalized quantum Arthur-Merlin proof systems. Section 4 provides a sketch of a proof of the qq-QAM-completeness of the CITM problem. Section 5 then proves Theorem 1.1, the collapse theorem for qq-QAM. This essentially shows the qq-QAM-hardness of the MAXOUTQEA problem also. Section 6 treats the inclusion of the standard QAM in qq-QAM<sub>1</sub> (Theorem 1.4). Section 7 proves Theorem 1.7, the complete classification of the complexity classes derived from generalized quantum Arthur-Merlin proof systems. Finally, Section 8 concludes the paper with some open problems. A proof of the MAXOUTQEA problem being in qq-QAM is provided in the appendix, which completes the proof of the qq-QAM-completeness of MAXOUTQEA (Theorem 1.3). Some of the technical proofs are relegated to the full version [19] of this paper.

## 2 Preliminaries

Throughout this paper, let  $\mathbb{N}$  and  $\mathbb{Z}^+$  denote the sets of positive and nonnegative integers, respectively, and let  $\Sigma = \{0, 1\}$  denote the binary alphabet set. A function  $f: \mathbb{Z}^+ \rightarrow \mathbb{N}$  is *polynomially bounded* if there exists a polynomial-time deterministic Turing machine that outputs  $1^{f(n)}$  on input  $1^n$ . A function  $f: \mathbb{Z}^+ \rightarrow [0, 1]$  is *negligible* if, for any polynomially bounded function  $g: \mathbb{Z}^+ \rightarrow \mathbb{N}$ , the inequality  $f(n) < 1/g(n)$  holds for all but finitely many values of  $n$ .

**Quantum fundamentals.** We assume the reader is familiar with the quantum formalism, including pure and mixed quantum states, density operators, and measurements, as well as the quantum circuit model (see Refs. [25, 17, 37], for instance). Some notations and properties are summarized here for later use.

For each  $k$  in  $\mathbb{N}$ , let  $\mathbb{C}(\Sigma^k)$  denote the  $2^k$ -dimensional complex Hilbert space whose standard basis vectors are indexed by the elements in  $\Sigma^k$ . In this paper, all Hilbert spaces are complex and have dimension a power of two. For a Hilbert space  $\mathcal{H}$ , let  $\mathbf{L}(\mathcal{H})$  denote the set of linear operators over  $\mathcal{H}$  (i.e., the set of linear mappings from  $\mathcal{H}$  to itself), and let  $\mathbf{D}(\mathcal{H})$  denote the set of density operators over  $\mathcal{H}$ . For Hilbert spaces  $\mathcal{H}$  and  $\mathcal{K}$ , let  $\mathbf{C}(\mathcal{H}, \mathcal{K})$  denote the set of quantum channels from  $\mathbf{D}(\mathcal{H})$  to  $\mathbf{D}(\mathcal{K})$  (i.e., the set of linear mappings from  $\mathbf{L}(\mathcal{H})$  to  $\mathbf{L}(\mathcal{K})$  that are completely positive and trace-preserving). As usual, let

$$|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$

denote the two-qubit state in  $\mathbb{C}(\Sigma^2)$  that forms an EPR pair, and let

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

denote the Pauli operators. For convenience, we may identify a unitary operator with the unitary transformation it induces. In particular, for a unitary operator  $U$ , the induced unitary transformation is also denoted by  $U$ .

For a linear operator  $A$ , the *trace norm* of  $A$  is defined by

$$\|A\|_{\text{tr}} = \text{tr} \sqrt{A^\dagger A}.$$

For a Hilbert space  $\mathcal{H}$  and two quantum states  $\rho$  and  $\sigma$  in  $\mathbf{D}(\mathcal{H})$ , the *trace distance* between  $\rho$  and  $\sigma$  is defined by

$$D(\rho, \sigma) = \frac{1}{2} \|\rho - \sigma\|_{\text{tr}}.$$

For Hilbert spaces  $\mathcal{H}$  and  $\mathcal{K}$  and two quantum channels  $\Phi$  and  $\Psi$  in  $\mathbf{C}(\mathcal{H}, \mathcal{K})$ , the *minimum output trace distance* between  $\Phi$  and  $\Psi$  is defined by

$$D_{\min}(\Phi, \Psi) = \min \{D(\Phi(\rho), \Psi(\sigma)) : \rho, \sigma \in \mathbf{D}(\mathcal{H})\}.$$

The minimum output trace distance satisfies the following property. The proof is found in the full version [19] of this paper.

► **Lemma 2.1.** *For any Hilbert spaces  $\mathcal{H}$  and  $\mathcal{K}$ , any quantum channels  $\Phi$  and  $\Psi$  in  $\mathbf{C}(\mathcal{H}, \mathcal{K})$ , and any  $k$  in  $\mathbb{N}$ ,*

$$1 - [1 - (D_{\min}(\Phi, \Psi))^2]^{\frac{k}{2}} \leq D_{\min}(\Phi^{\otimes k}, \Psi^{\otimes k}) \leq k D_{\min}(\Phi, \Psi).$$

For any quantum state  $\rho$ , the *von Neumann entropy* of  $\rho$  is defined by

$$S(\rho) = -\text{tr}(\rho \log \rho).$$

A special case of the von Neumann entropy is the *Shannon entropy* of a probability distribution  $\mu$ , which is defined by

$$H(\mu) = S(\mu)$$

by viewing probability distributions as special cases of quantum states with diagonal density operators.

For Hilbert spaces  $\mathcal{H}$  and  $\mathcal{K}$  and a quantum channel  $\Phi$  in  $\mathbf{C}(\mathcal{H}, \mathcal{K})$ , the *maximum output von Neumann entropy* of  $\Phi$  is defined by

$$S_{\max}(\Phi) = \max \{S(\Phi(\rho)) : \rho \in \mathbf{D}(\mathcal{H})\}.$$

This paper uses the following two properties on von Neumann entropy.

The first lemma provides an upper bound on the von Neumann entropy of a mixture of quantum states [25, Theorem 11.10].

► **Lemma 2.2.** *For any Hilbert space  $\mathcal{H}$  and any quantum state  $\rho$  in  $\mathbf{D}(\mathcal{H})$  such that  $\rho = \sum_j \mu_j \rho_j$  for some probability distribution  $\mu = \{\mu_j\}$  and quantum states  $\rho_j$  in  $\mathbf{D}(\mathcal{H})$ ,*

$$S(\rho) \leq H(\mu) + \sum_j \mu_j S(\rho_j).$$

The second lemma describes relations between the von Neumann entropy of a quantum state and the trace distance between the state and the totally mixed state (a similar statement appeared in the full version of Ref. [8] without a proof). The proof of the statement described here is found in the full version [19] of this paper.

► **Lemma 2.3.** *For any quantum state  $\rho$  of  $n$  qubits, it holds that*

$$(1 - D(\rho, (I/2)^{\otimes n}) - 2^{-n})n \leq S(\rho) \leq n - \log \frac{1}{1 - D(\rho, (I/2)^{\otimes n})} + 2.$$

**Quantum circuits.** Following conventions, this paper defines quantum Arthur-Merlin proof systems in terms of quantum circuits. In particular, this paper uses the following notion of polynomial-time uniformly generated families of quantum circuits.

A *quantum circuit* is specified by a series of quantum gates, each of which is applied to some designated set of qubits. It is assumed that any quantum circuit is composed of gates in some reasonable, universal, finite set of quantum gates. A *description* of a quantum circuit is a string in  $\Sigma^*$  that encodes the specification of the quantum circuit. The encoding must be a “natural” one, i.e., the number of gates in a circuit encoded is not more than the length of the description of that circuit, and each gate of the circuit is specifiable by a deterministic procedure in time polynomial with respect to the length of the description.

A family  $\{Q_x\}_{x \in \Sigma^*}$  of quantum circuits is *polynomial-time uniformly generated* if there exists a polynomial-time deterministic procedure that, on input  $x$  in  $\Sigma^*$ , outputs a description of  $Q_x$ . For convenience, we may identify a circuit  $Q_x$  with the unitary operator it induces.

For the results in which perfect completeness is concerned, this paper assumes a gate set with which the Hadamard and any classical reversible transformations can be exactly implemented. Note that this assumption is satisfied by many standard gate sets such as the Shor basis [31] consisting of the Hadamard,  $i$ -phase-shift, and Toffoli gates, and the gate set consisting of the Hadamard, Toffoli, and NOT gates [30, 2]. Moreover, as the Hadamard transformation in a sense can be viewed as a quantum analogue of the classical operation of flipping a fair coin, our assumption would be the most natural quantum correspondence to the tacit classical assumption in randomized complexity theory that fair coins and perfect logical gates are available. Hence, the authors believe that the condition above is very reasonable and not restrictive.

Since non-unitary and unitary quantum circuits are equivalent in computational power [3], it is sufficient to treat only unitary quantum circuits, as defined above. Nevertheless, for readability, most procedures in this paper will be described using intermediate projective measurements and unitary operations conditioned on the outcome of the measurements. All of these intermediate measurements can be deferred to the end of the procedure by a standard technique so that the procedure becomes implementable with a unitary circuit.

### 3 Generalized quantum Arthur-Merlin proof systems

A generalized quantum Arthur-Merlin proof system consists of a polynomial-time quantum verifier and an all-powerful quantum prover. For any constant  $m \geq 1$  and any message-type  $t_j$  in  $\{c, q\}$  for each  $j$  in  $\{1, \dots, m\}$ , a generalized quantum Arthur-Merlin proof system is of  $t_m \cdots t_1$ -QAM type if the message at the  $(m - j + 1)$ st turn is quantum (resp. is restricted to classical) for each  $j$  such that  $t_j = q$  (resp.  $t_j = c$ ).

Formally, an  $m$ -turn quantum verifier  $V$  for generalized quantum Arthur-Merlin proof systems is a polynomial-time computable mapping of the form  $V: \Sigma^* \rightarrow \Sigma^*$ . For each  $x$  in  $\Sigma^*$ ,  $V(x)$  is interpreted as a description of a quantum circuit acting on  $(q_V(|x|) + m q_M(|x|))$  qubits with a specification of a  $q_V(|x|)$ -qubit quantum register  $V$  and each  $q_M(|x|)$ -qubit quantum register  $M_j$  for  $j$  in  $\{1, \dots, m\}$ , for some polynomially bounded functions  $q_V, q_M: \mathbb{Z}^+ \rightarrow \mathbb{N}$ . One of the qubits in  $V$  is designated as the output qubit. At the  $(m - j + 1)$ st turn for any even  $j$  such that  $2 \leq j \leq m - 1$ ,  $V$  receives a message from a prover, either classical or quantum, which is stored in the quantum register  $M_{m-j}$ . When the system is of  $t_m \cdots t_1$ -QAM type, at the  $(m - j + 1)$ st turn for any even  $j$  such that  $2 \leq j \leq m$ , if  $t_j = c$ ,  $V$  flips a fair coin  $q_M(|x|)$  times to obtain a binary string  $r$  of length  $q_M(|x|)$ , then sends  $r$  to the prover, and stores  $r$  in the quantum register  $M_{m-j+1}$ , while if  $t_j = q$ ,  $V$  generates  $q_M(|x|)$  EPR

pairs  $|\Phi^+\rangle^{\otimes q_M(|x|)}$ , then sends the second halves of them to the prover, and stores the first halves of them in  $M_{m-j+1}$ . Upon receiving a message at the  $m$ th turn from the prover, either classical or quantum, which is stored in the quantum register  $M_m$ ,  $V$  prepares the  $q_V(|x|)$ -qubit quantum register  $V$ , all the qubits of which are initialized to the  $|0\rangle$  state.  $V$  then performs the final verification procedure by applying the circuit  $V(x)$  to  $(V, M_1, \dots, M_m)$  and then measuring the output qubit in the computational basis, where the outcome  $|1\rangle$  is interpreted as “accept”, and the outcome  $|0\rangle$  is interpreted as “reject”.

Similarly, an  $m$ -turn quantum prover  $P$  for generalized quantum Arthur-Merlin proof systems is a mapping from  $\Sigma^*$  to a sequence of  $\lceil m/2 \rceil$  unitary transformations with a specification of quantum registers they acts on. No restrictions are placed on the complexity of  $P$ . For each  $x$  in  $\Sigma^*$ ,  $P(x)$  is interpreted as a sequence of  $\lceil m/2 \rceil$  unitary transformations  $P(x)_{2\lceil m/2 \rceil-1}, \dots, P(x)_3, P(x)_1$  acting on  $(q_M(|x|) + q_P(|x|))$  qubits with a specification of a  $q_P(|x|)$ -qubit quantum register  $P$ , for some polynomially bounded function  $q_M: \mathbb{Z}^+ \rightarrow \mathbb{N}$  and some function  $q_P: \mathbb{Z}^+ \rightarrow \mathbb{N}$ . At the beginning of the protocol,  $P$  prepares the  $q_P(|x|)$ -qubit quantum register  $P$  (and a  $q_M(|x|)$ -qubit quantum register  $M_1$  also, if  $m$  is odd). Without loss of generality, one can assume that all the qubits in  $P$  (and in  $M_1$  when  $P$  prepares it) are initialized to the  $|0\rangle$  state at the beginning of the protocol. At the  $(m-j+1)$ st turn for any odd  $j$  such that  $1 \leq j \leq m-1$ ,  $P$  receives a message from the verifier, either classical or quantum, which is stored in the quantum register  $M_{m-j+1}$ . When the system is of  $t_m \cdots t_1$ -QAM type, at the  $(m-j+1)$ st turn for any odd  $j$  such that  $1 \leq j \leq m$ ,  $P$  applies  $P(x)_j$  to  $(M_{m-j+1}, P)$ . If  $t_j = c$ ,  $P$  further measures each qubit in  $M_{m-j+1}$  in the computational basis.  $P$  then sends  $M_{m-j+1}$  to the verifier.

An  $m$ -turn generalized quantum Arthur-Merlin proof system  $\Pi$  is then specified by each message-type  $t_j$  in  $\{c, q\}$  for  $j$  in  $\{1, \dots, m\}$  and an  $m$ -turn quantum verifier  $V$  for generalized quantum Arthur-Merlin proof systems. An  $m$ -turn quantum prover  $P$  for  $t_m \cdots t_1$ -QAM-type systems is *compatible* with  $\Pi$  if the function  $q_M$  of  $P$  is the same as that of  $V$ . In what follows, provers are always assumed to be compatible. For any generalized quantum Arthur-Merlin proof system  $\Pi$ , let  $\text{MAP}_x(\Pi)$  denote the *maximum acceptance probability* in  $\Pi$  on input  $x$  in  $\Sigma^*$ , which is the maximum of the acceptance probability of the verifier in  $\Pi$  on input  $x$  over all quantum provers compatible with  $\Pi$ . The complexity class  $t_m \cdots t_1$ -QAM( $m, c, s$ ) derived from generalized quantum Arthur-Merlin proof systems of  $t_m \cdots t_1$ -QAM type, with completeness  $c$  and soundness  $s$ , is defined as follows.

► **Definition 3.1.** Given a constant  $m$  in  $\mathbb{N}$ , functions  $c, s: \mathbb{Z}^+ \rightarrow [0, 1]$  satisfying  $c > s$ , and each message-type  $t_j$  in  $\{c, q\}$  for  $j$  in  $\{1, \dots, m\}$ , a promise problem  $A = (A_{\text{yes}}, A_{\text{no}})$  is in  $t_m \cdots t_1$ -QAM( $m, c, s$ ) if there exists an  $m$ -turn quantum verifier  $V$  for generalized quantum Arthur-Merlin proof systems, such that, for the  $t_m \cdots t_1$ -QAM-type proof system  $\Pi$  specified by  $V$  and for every input  $x$  in  $\Sigma^*$ ,

**(Completeness)** if  $x$  is in  $A_{\text{yes}}$ ,  $\text{MAP}_x(\Pi)$  is at least  $c(|x|)$ , and

**(Soundness)** if  $x$  is in  $A_{\text{no}}$ ,  $\text{MAP}_x(\Pi)$  is at most  $s(|x|)$ .

Using this definition, the classes  $t_m \cdots t_1$ -QAM( $m$ ) and  $t_m \cdots t_1$ -QAM<sub>1</sub>( $m$ ) of problems having generalized quantum Arthur-Merlin proof systems of  $t_m \cdots t_1$ -QAM type with two-sided bounded error, and those with one-sided bounded error of perfect completeness, respectively, are defined as follows.

► **Definition 3.2.** Given a constant  $m$  in  $\mathbb{N}$  and each message-type  $t_j$  in  $\{c, q\}$  for  $j$  in  $\{1, \dots, m\}$ , a promise problem  $A = (A_{\text{yes}}, A_{\text{no}})$  is in  $t_m \cdots t_1$ -QAM( $m$ ) iff  $A$  is in  $t_m \cdots t_1$ -QAM( $m, 1 - \varepsilon, \varepsilon$ ) for some negligible function  $\varepsilon: \mathbb{Z}^+ \rightarrow [0, 1]$ .



► **Definition 3.3.** Given a constant  $m$  in  $\mathbb{N}$  and each message-type  $t_j$  in  $\{c, q\}$  for  $j$  in  $\{1, \dots, m\}$ , a promise problem  $A = (A_{\text{yes}}, A_{\text{no}})$  is in  $t_m \cdots t_1$ -QAM $_1(m)$  iff  $A$  is in  $t_m \cdots t_1$ -QAM( $m, 1, \varepsilon$ ) for some negligible function  $\varepsilon: \mathbb{Z}^+ \rightarrow [0, 1]$ .

In the case where the number of turns is clear, the parameter  $m$  may be omitted, e.g., ccqq-QAM(4) may be abbreviated as ccqq-QAM.

Similar to general quantum interactive proof systems, the perfect parallel repetition theorem holds for generalized quantum Arthur-Merlin proof systems.

► **Lemma 3.4.** *For any generalized quantum Arthur-Merlin proof system  $\Pi$ , for any  $k$  in  $\mathbb{N}$  and the generalized quantum Arthur-Merlin proof system  $\Pi^{\otimes k}$  resulting from the  $k$ -fold parallel repetition of  $\Pi$ , and for every input  $x$  in  $\Sigma^*$ , it holds that*

$$\text{MAP}_x(\Pi^{\otimes k}) = (\text{MAP}_x(\Pi))^k.$$

**Proof.** Fix any number  $m$  of turns and any message-types  $t_1, \dots, t_m$  in  $\{c, q\}$ . For any proof system  $\Pi$  of  $t_m \cdots t_1$ -QAM type, let  $Q(\Pi)$  be the  $m$ -turn (general) quantum interactive proof system that exactly simulates  $\Pi$  as follows: on every input  $x$  in  $\Sigma^*$ , the verifier in  $Q(\Pi)$  behaves exactly in the same manner as Arthur in  $\Pi$  except that, upon receiving the  $j$ th message from a prover (resp. sending the  $j$ th message to a prover), if  $t_j = c$  in  $\Pi$ , the verifier of  $Q(\Pi)$  first makes sure that the received message (resp. the sent message) is indeed classical by taking a copy of the message by CNOT operations (and the copied message will never be touched in the rest of the protocol). Clearly, it is meaningless for a malicious prover in  $Q(\Pi)$  to send a quantum message when the original message-type was classical in  $\Pi$ . Therefore, for every input  $x$ , the maximum acceptance probability in  $Q(\Pi)$  is exactly  $\text{MAP}_x(\Pi)$ . Now from the perfect parallel repetition theorem for general quantum interactive proofs [11], the  $k$ -fold parallel repetition  $(Q(\Pi))^{\otimes k}$  of  $Q(\Pi)$  has its maximum acceptance probability exactly  $(\text{MAP}_x(\Pi))^k$  for every  $x$ . As the proof system  $(Q(\Pi))^{\otimes k}$  is identical to the  $m$ -turn (general) quantum interactive proof system  $Q(\Pi^{\otimes k})$  that exactly simulates the proof system  $\Pi^{\otimes k}$  of  $t_m \cdots t_1$ -QAM type that is the  $k$ -fold parallel repetition of  $\Pi$ , it holds that  $\text{MAP}_x(\Pi^{\otimes k}) = (\text{MAP}_x(\Pi))^k$  for every  $x$ , as claimed. ◀

Using Lemma 3.4, one can show the following amplification properties on generalized quantum Arthur-Merlin proof systems, which ensure that Definitions 3.2 and 3.3 give a robust definition in terms of completeness and soundness parameters.

► **Lemma 3.5.** *For any constant  $m$  in  $\mathbb{N}$ , any message-types  $t_1, \dots, t_m$  in  $\{c, q\}$ , any polynomially bounded function  $p: \mathbb{Z}^+ \rightarrow \mathbb{N}$ , and any polynomial-time computable functions  $c, s: \mathbb{Z}^+ \rightarrow [0, 1]$  satisfying  $c - s \geq \frac{1}{q}$  for some polynomially bounded function  $q: \mathbb{Z}^+ \rightarrow \mathbb{N}$ ,*

$$t_m \cdots t_1\text{-QAM}(m, c, s) \subseteq t_m \cdots t_1\text{-QAM}(m, 1 - 2^{-p}, 2^{-p}).$$

► **Lemma 3.6.** *For any constant  $m$  in  $\mathbb{N}$ , any message-types  $t_1, \dots, t_m$  in  $\{c, q\}$ , any polynomially bounded function  $p: \mathbb{Z}^+ \rightarrow \mathbb{N}$ , and any polynomial-time computable function  $s: \mathbb{Z}^+ \rightarrow [0, 1]$  satisfying  $1 - s \geq \frac{1}{q}$  for some polynomially bounded function  $q: \mathbb{Z}^+ \rightarrow \mathbb{N}$ ,*

$$t_m \cdots t_1\text{-QAM}(m, 1, s) \subseteq t_m \cdots t_1\text{-QAM}(m, 1, 2^{-p}).$$

**Proofs of Lemmas 3.5 and 3.6 (Sketch).** Lemma 3.6 is immediate from Lemma 3.4 by considering a parallel repetition of an appropriately many number of times.

To prove Lemma 3.5, as in Refs. [1, 21, 14], one first makes the completeness exponentially close to one, while keeping the soundness bounded away from one, by performing a sufficiently



---

**Verifier's qq-QAM Protocol for CITM( $a, b$ )**

1. Prepare  $q_{\text{out}}$ -qubit registers  $S_1$  and  $S_2$ , and generate  $q_{\text{out}}$  EPR pairs  $|\Phi^+\rangle^{\otimes q_{\text{out}}}$  in  $(S_1, S_2)$  so that the  $j$ th qubit of  $S_1$  and that of  $S_2$  form an EPR pair, for every  $j$  in  $\{1, \dots, q_{\text{out}}\}$ . Send  $S_2$  to the prover.
  2. Receive a  $(q_{\text{all}} - q_{\text{out}})$ -qubit quantum register  $R$  from the prover. Apply the unitary transformation  $U_{Q_x}^\dagger$  to  $(R, S_1)$ . Accept if all the qubits in  $A$  are in state  $|0\rangle$ , and reject otherwise, where  $A$  is the quantum register consisting of the last  $(q_{\text{all}} - q_{\text{in}})$  qubits of  $(R, S_1)$  (i.e., the non-input qubits of  $Q_x$ ).
- 

■ **Figure 1** Verifier's qq-QAM protocol for CITM.

many number of attempts of a given system in parallel and accepting only when a reasonably large fraction of the attempts results in acceptance. Lemma 3.5 is then immediate from Lemma 3.4 by running this system of almost-perfect completeness in parallel appropriately many times. The rigorous proof is found in the full version [19] of this paper. ◀

#### 4 qq-QAM-completeness of CITM

This section proves Theorem 1.2, which states that the CITM problem is complete for the class qq-QAM.

First, it is proved that  $\text{CITM}(a, b)$  is in qq-QAM for appropriately chosen parameters  $a$  and  $b$ . The proof is a special case of the proof of the CLOSE IMAGE problem being in QIP(2) [32, 12].

► **Lemma 4.1.** *For any constants  $a$  and  $b$  in  $[0, 1]$  satisfying  $(1 - a)^2 > 1 - b^2$ ,  $\text{CITM}(a, b)$  is in qq-QAM.*

**Proof (Sketch).** Let  $Q_x$  be a quantum circuit of an instance  $x$  of  $\text{CITM}(a, b)$  acting on  $q_{\text{all}}$  qubits with  $q_{\text{in}}$  specified input qubits and  $q_{\text{out}}$  specified output qubits. Without loss of generality, one can assume that the first  $q_{\text{in}}$  qubits correspond to the input qubits, and the last  $q_{\text{out}}$  qubits correspond to the output qubits. Let  $U_{Q_x}$  denote the unitary operator induced by  $Q_x$ . We construct a verifier  $V$  of the qq-QAM proof system with completeness  $(1 - a)^2$  and soundness  $1 - b^2$  as follows (recall that  $a$  and  $b$  are constants in the interval  $[0, 1]$  such that  $(1 - a)^2 > 1 - b^2$ , and thus this qq-QAM proof system is sufficient for the claim).

Let  $S_1$  and  $S_2$  be quantum registers of  $q_{\text{out}}$  qubits. The verifier  $V$  first generates  $q_{\text{out}}$  EPR pairs  $|\Phi^+\rangle^{\otimes q_{\text{out}}}$  in  $(S_1, S_2)$  so that the  $j$ th qubit of  $S_1$  and that of  $S_2$  form an EPR pair, for every  $j$  in  $\{1, \dots, q_{\text{out}}\}$ . Then  $V$  sends  $S_2$  to the prover. Upon receiving a quantum register  $R$  of  $(q_{\text{all}} - q_{\text{out}})$  qubits,  $V$  applies the unitary transformation  $U_{Q_x}^\dagger$  to  $(R, S_1)$ . Letting  $A$  be the quantum register consisting of the last  $(q_{\text{all}} - q_{\text{in}})$  qubits of the register  $(R, S_1)$  (i.e., corresponding to the *non-input* qubits of  $Q_x$ ),  $V$  accepts  $x$  if and only if all the qubits in  $A$  are in state  $|0\rangle$ . Figure 1 summarizes the protocol of the verifier  $V$ .

The claim follows from a rigorous analysis of this protocol, which is relegated to the full version [19] of this paper. ◀

Now the CITM problem is proved to be hard for qq-QAM.

► **Lemma 4.2.** *For any constants  $a$  and  $b$  satisfying  $0 < a < b < 1$ ,  $\text{CITM}(a, b)$  is hard for qq-QAM under polynomial-time many-one reduction.*

---

**Algorithm Corresponding to Quantum Circuit  $Q_x$** 

1. Prepare quantum registers  $V$  and  $M$ , each of  $q_V$  and  $q_M$  qubits, respectively. Denote by  $S$  and  $\bar{S}$  the quantum registers consisting of the last  $q_S$  and first  $(q_V - q_S)$  qubits of  $V$ , respectively. The last  $(q_S + q_M)$  qubits of  $(V, M) = (\bar{S}, S, M)$  (i.e., all the qubits in  $(S, M)$ ) are designated as the input qubits, while the last  $q_S$  qubits of  $V = (\bar{S}, S)$  (i.e., all the qubits in  $S$ ) are designated as the output qubits.
  2. Flip a fair coin, and proceed to Step 2.a if it results in “Heads”, and proceed to Step 2.b if it results in “Tails”.
    - a. Output all the qubits in  $S$ .
    - b. Perform  $V_x$  over  $(V, M) = (\bar{S}, S, M)$ . If the first qubit of  $V$  is in state  $|1\rangle$ , output the totally mixed state  $(I/2)^{\otimes q_S}$  (by first generating the totally mixed state using fresh ancillae, and then swapping the qubits in  $S$  with the generated totally mixed state), and output  $|0\rangle^{\otimes q_S}$  otherwise (by swapping the qubits in  $S$  with  $q_S$  fresh ancillae).
- 

■ **Figure 2** The construction of the quantum circuit  $Q_x$ .

**Proof (Sketch).** Let  $A = (A_{\text{yes}}, A_{\text{no}})$  be a problem in qq-QAM. Then  $A$  has a qq-QAM proof system with completeness  $c$  and soundness  $s$  for some appropriately chosen constants  $c$  and  $s$  satisfying  $0 < s < c < 1$ . Let  $V$  be the quantum verifier witnessing this proof system. Fix an input  $x$ , and let  $V$  and  $M$  be quantum registers consisting of  $q_V$  and  $q_M$  qubits, respectively, where  $V$  corresponds to the private qubits of  $V$  and  $M$  corresponds to the message qubits  $V$  would receive on input  $x$ . Without loss of generality, one can assume that the first qubit of  $V$  is the output qubit of  $V$ , and the last  $q_S$  qubits of  $V$  form the quantum register  $S$  corresponding to the halves of the EPR pairs  $V$  would keep until the final verification procedure is performed. Let  $\bar{S}$  be the quantum register of  $(q_V - q_S)$  qubits consisting of the first  $(q_V - q_S)$  qubits of  $V$  (i.e., all the private qubits of  $V$  but those belonging to  $S$ ). Denote by  $V_x$  the unitary operator induced by this  $V$  on input  $x$ .

We construct a quantum circuit  $Q_x$  that exactly implements the following algorithm. The circuit  $Q_x$  expects to receive a  $(q_S + q_M)$ -qubit state as its input, and prepares the quantum registers  $V = (\bar{S}, S)$  and  $M$ , where the input state is expected to be stored in  $(S, M)$ . Then with probability one-half,  $Q_x$  just outputs the state in the register  $S$ . Otherwise  $Q_x$  performs  $V_x$  over  $(V, M) = (\bar{S}, S, M)$ , and outputs the totally mixed state  $(I/2)^{\otimes q_S}$  if the first qubit of  $V$  is in state  $|1\rangle$  (i.e., if the system is in an accepting state of the original verifier  $V$ ), and outputs  $(|0\rangle\langle 0|)^{\otimes q_S}$  if the first qubit of  $V$  is in state  $|0\rangle$  (i.e., if the system is in a rejecting state of the original verifier  $V$ ). Figure 2 summarizes the construction of the circuit  $Q_x$ .

The qq-QAM-hardness of  $\text{CITM}(a, 1/20)$  for any positive constant  $a < 1/20$  follows from a rigorous analysis of the properties of this circuit by appropriately choosing  $c$  and  $s$ , which is found in the full version [19] of this paper. The qq-QAM-hardness of  $\text{CITM}(a, b)$  for any constants  $a$  and  $b$  satisfying  $0 < a < b < 1$  then follows from Lemma 2.1 by first creating an instance  $Q_x$  of  $\text{CITM}(a/k, 1/20)$  according to the construction above, for  $k = \lceil 2^{\frac{\ln(1/(1-b))}{\ln(400/399)}} \rceil$ , and then constructing another circuit  $Q'_x$  that places  $k$  copies of  $Q_x$  in parallel. ◀

From Lemmas 4.1 and 4.2, Theorem 1.2 follows. Note that, with essentially the same proofs as those of Lemmas 4.1 and 4.2, one can show that for any  $b$  in  $(0, 1)$ ,  $\text{CITM}(0, b)$  is in qq-QAM<sub>1</sub> and is hard for qq-QAM<sub>1</sub>, and thus, the following corollary holds.

► **Corollary 4.3.** *For any constant  $b$  in  $(0, 1)$ ,  $\text{CITM}(0, b)$  is qq-QAM<sub>1</sub>-complete under polynomial-time many-one reduction.*

► **Remark.** The proofs of Lemmas 4.1 and 4.2 actually also show that the variant of the CITM problem where the number of output qubits of the circuit is a fixed constant independent of instances is complete for the class  $\text{QMA}^{\text{const-EPR}}$  introduced in Ref. [20], and thus, it is QMA-complete since  $\text{QMA}^{\text{const-EPR}} = \text{QMA}$  [5].

## 5 Collapse theorem for qq-QAM

This section proves Theorem 1.1, the quantum analogue of Babai's collapse theorem [4] stating that  $c \cdots \text{cqq-QAM}(m) = \text{qq-QAM}$  for any constant  $m \geq 2$ .

First, it is proved that for any constant  $m \geq 4$ ,  $c \cdots \text{cqq-QAM}(m) \subseteq \text{ccqq-QAM}$  holds, meaning that the first  $(m - 4)$  classical turns can be removed. The proof essentially relies on the observation that the techniques used in the classical result by Babai [4] can be applied to the quantum setting as well.

► **Lemma 5.1.** *For any constant  $m \geq 4$ ,  $c \cdots \text{cqq-QAM}(m) \subseteq \text{ccqq-QAM}$ .*

**Proof.** It suffices to show that  $c \cdots \text{cqq-QAM}(m) \subseteq c \cdots \text{cqq-QAM}(m - 1)$  for any odd constant  $m \geq 5$ , and  $c \cdots \text{cqq-QAM}(m) \subseteq c \cdots \text{cqq-QAM}(m - 2)$  for any even constant  $m \geq 6$ .

Let  $A = (A_{\text{yes}}, A_{\text{no}})$  be a problem in  $c \cdots \text{cqq-QAM}(m)$ . By Lemma 3.5,  $A$  has an  $m$ -turn  $c \cdots \text{cqq-QAM}$  proof system  $\Pi$  with completeness  $1 - 2^{-8}$  and soundness  $2^{-8}$ . Without loss of generality, one can assume that, for every input of length  $n$ , every classical message exchanged consists of  $l(n)$  bits for some polynomially bounded function  $l: \mathbb{Z}^+ \rightarrow \mathbb{N}$ .

First consider the case with odd  $m$ , where the first turn is for the prover. Fix an input  $x$  in  $\Sigma^*$ , and let  $w_x(y, r)$  be the maximum of the probability that the prover can make the verifier accept, under the condition that the first message from the prover is  $y$  in  $\Sigma^{l(|x|)}$  and the second message from the verifier is  $r$  in  $\Sigma^{l(|x|)}$ . Then, the maximum acceptance probability in  $\Pi$  is given by  $\text{MAP}_x(\Pi) = \max_{y \in \Sigma^{l(|x|)}} \{E[w_x(y, r)]\}$ , where the expectation is taken over the uniform distribution with respect to  $r$  in  $\Sigma^{l(|x|)}$ . Note that  $\text{MAP}_x(\Pi) \geq 1 - 2^{-8}$  if  $x$  is in  $A_{\text{yes}}$ , and  $\text{MAP}_x(\Pi) \leq 2^{-8}$  if  $x$  is in  $A_{\text{no}}$ .

Consider the  $(m - 1)$ -turn  $c \cdots \text{cqq-QAM}$  proof system  $\Pi'$  specified by the following protocol of the verifier: At the first turn, the verifier sends  $k(|x|)$  strings  $r_1, \dots, r_{k(|x|)}$  chosen uniformly at random from  $\Sigma^{l(|x|)}$ , for some polynomially bounded function  $k: \mathbb{Z}^+ \rightarrow \mathbb{N}$ . Upon receiving a string  $y$  in  $\Sigma^{l(|x|)}$  and  $k(|x|)$  strings  $z_1, \dots, z_{k(|x|)}$  in  $\Sigma^{l(|x|)}$  at the third turn, the verifier enters the simulations of the last  $(m - 3)$  turns of communications of  $\Pi$ , by running in parallel  $k(|x|)$  attempts of such simulations, where the  $j$ th attempt assumes that the first three messages in the original proof system  $\Pi$  were  $y$ ,  $r_j$ , and  $z_j$ , respectively, for each  $j$  in  $\{1, \dots, k(|x|)\}$ . The verifier accepts if and only if more than  $k(|x|)/2$  attempts result in acceptance in these simulations of  $\Pi$ . Figure 3 summarizes the protocol of this verifier in  $\Pi'$ .

In fact, the construction of this proof system  $\Pi'$  is exactly the same as in Ref. [4] except that the last two messages exchanged are quantum and the final verification of the verifier is a polynomial-time quantum computation in the present case. The analysis in Ref. [4] works also in the present case, since it only relies on the fact that  $w_x(y, r)$  gives the conditional probability defined above, and from Lemma 3.4, the perfect parallel repetition theorem holds for general quantum Arthur-Merlin proof systems. In particular, the following property holds also in the present case (see Lemmas 3.3 and 3.4 of Ref. [4]).

► **Claim 1.**  $1 - 2^{k(|x|)}(1 - \text{MAP}_x(\Pi))^{k(|x|)/2} \leq \text{MAP}_x(\Pi') \leq 2^{k(|x|)+l(|x|)}(\text{MAP}_x(\Pi))^{k(|x|)/2}$ .

---

**Verifier's Protocol for Reducing the Number of Turns by One (for Odd  $m$ )**


---

1. Send  $k(|x|)$  strings  $r_1, \dots, r_{k(|x|)}$ , each chosen uniformly at random from  $\Sigma^{l(|x|)}$ , to the prover, for some polynomially bounded function  $k: \mathbb{Z}^+ \rightarrow \mathbb{N}$ .
  2. Receive a string  $y$  in  $\Sigma^{l(|x|)}$  and  $k(|x|)$  strings  $z_1, \dots, z_{k(|x|)}$  in  $\Sigma^{l(|x|)}$  from the prover. Run in parallel  $k(|x|)$  attempts of the  $(m-3)$ -turn protocol that simulates the last  $(m-3)$  turns of communications of the original  $m$ -turn  $c \cdots$  cqq-QAM proof system  $\Pi$  on input  $x$ , where the  $j$ th attempt assumes that the first three messages in  $\Pi$  were  $y, r_j$ , and  $z_j$ , respectively, for each  $j$  in  $\{1, \dots, k(|x|)\}$ . Accept if more than  $k(|x|)/2$  attempts result in acceptance in these simulations of  $\Pi$ , and reject otherwise.
- 

■ **Figure 3** Verifier's protocol in  $\Pi'$  for reducing the number of turns by one when  $m$  is odd.

Now let  $k = \lceil \frac{2+l}{3} \rceil$ . If  $x$  is in  $A_{\text{yes}}$ , then  $\text{MAP}_x(\Pi')$  is at least

$$1 - 2^{k(|x|)}(1 - \text{MAP}_x(\Pi))^{k(|x|)/2} \geq 1 - 2^{k(|x|)}(2^{-8})^{k(|x|)/2} \geq 1 - \frac{1}{2^{l(|x|)+2}} \geq \frac{3}{4},$$

while if  $x$  is in  $A_{\text{no}}$ , then  $\text{MAP}_x(\Pi')$  is at most

$$2^{k(|x|)+l(|x|)}(\text{MAP}_x(\Pi))^{k(|x|)/2} \leq 2^{k(|x|)+l(|x|)}(2^{-8})^{k(|x|)/2} \leq \frac{1}{4},$$

which completes the proof for the case with odd  $m$ .

Next consider the case with even  $m$ , where the first message is a random string from a verifier. Let  $\Pi^{(-1)}$  be the  $(m-1)$ -turn  $c \cdots$  cqq-QAM proof system that on input  $(x, r)$  simulates the last  $(m-1)$  turns of  $\Pi$  on  $x$  under the condition that the first message in  $\Pi$  was  $r$  in  $\Sigma^{l(|x|)}$ . Let  $B = (B_{\text{yes}}, B_{\text{no}})$  be the following promise problem in  $c \cdots$  cqq-QAM( $m-1$ ):

$$B_{\text{yes}} = \{(x, r): \text{MAP}_{(x,r)}(\Pi^{(-1)}) \geq 2/3\}, \quad B_{\text{no}} = \{(x, r): \text{MAP}_{(x,r)}(\Pi^{(-1)}) \leq 1/3\}.$$

Note that, if  $x$  is in  $A_{\text{yes}}$ , then  $(x, r)$  is in  $B_{\text{yes}}$  for at least  $(1 - 3 \cdot 2^{-8})$ -fraction of the choices of  $r$ . Similarly, if  $x$  is in  $A_{\text{no}}$ , then  $(x, r)$  is in  $B_{\text{no}}$  for at least  $(1 - 3 \cdot 2^{-8})$ -fraction of the choices of  $r$ . By the result for the case with odd  $m$  above, it holds that  $B$  is in  $c \cdots$  cqq-QAM( $m-2$ ). Thus, there exists an  $(m-2)$ -turn  $c \cdots$  cqq-QAM proof system  $\Pi^{(-2)}$  for  $B$  such that if  $(x, r)$  is in  $B_{\text{yes}}$ ,  $\text{MAP}_{(x,r)}(\Pi^{(-2)})$  is at least  $2/3$ , while if  $(x, r)$  is in  $B_{\text{no}}$ ,  $\text{MAP}_{(x,r)}(\Pi^{(-2)})$  is at most  $1/3$ . Note that the first turn of  $\Pi^{(-2)}$  is a turn for the verifier, and thus, one can merge the turn for sending  $r$  with the first turn of  $\Pi^{(-2)}$ . This results in an  $(m-2)$ -turn  $c \cdots$  cqq-QAM proof system  $\Pi''$  for  $A$  in which at the first turn the new verifier sends a string  $r$  in  $\Sigma^{l(|x|)}$  chosen uniformly at random in addition to the original first message of the verifier in  $\Pi^{(-2)}$  on input  $(x, r)$ , and then behaves exactly in the same manner as the verifier in  $\Pi^{(-2)}$  on input  $(x, r)$  in the rest of the protocol. If  $x$  is in  $A_{\text{yes}}$ ,  $\text{MAP}_x(\Pi'')$  is at least  $(1 - 3 \cdot 2^{-8}) \cdot (2/3) > 5/8$ , while if  $x$  is in  $A_{\text{no}}$ ,  $\text{MAP}_x(\Pi'')$  is at most  $3 \cdot 2^{-8} + (1 - 3 \cdot 2^{-8}) \cdot (1/3) < 3/8$ , which is sufficient for the claim, due to Lemma 3.5. ◀

Second, using the fact that CITM is qq-QAM-complete, it is proved that cqq-QAM is included in qq-QAM.

► **Lemma 5.2.**  $cqq\text{-QAM} \subseteq qq\text{-QAM}$ .

**Proof.** Let  $A = (A_{\text{yes}}, A_{\text{no}})$  be a problem in cqq-QAM. Then,  $A$  has a cqq-QAM proof system  $\Pi$  with completeness  $2/3$  and soundness  $1/3$ . Let  $l: \mathbb{Z}^+ \rightarrow \mathbb{N}$  be the polynomially bounded function that specifies the length of the first message in  $\Pi$ . Consider the qq-QAM

proof system  $\Pi^{\text{qq}}$  that on input  $(x, w)$  simulates the last two turns of  $\Pi$  on  $x$  under the condition that the first message in  $\Pi$  was  $w$  in  $\Sigma^{l(|x|)}$ . Let  $B = (B_{\text{yes}}, B_{\text{no}})$  be the following promise problem in qq-QAM:

$$B_{\text{yes}} = \{(x, w) : \text{MAP}_{(x,w)}(\Pi^{\text{qq}}) \geq 2/3\}, \quad B_{\text{no}} = \{(x, w) : \text{MAP}_{(x,w)}(\Pi^{\text{qq}}) \leq 1/3\}.$$

Note that for any  $x$ , if  $x$  is in  $A_{\text{yes}}$ , there exists a string  $w$  in  $\Sigma^{l(|x|)}$  such that  $(x, w)$  is in  $B_{\text{yes}}$ , and if  $x$  is in  $A_{\text{no}}$ , for every string  $w$  in  $\Sigma^{l(|x|)}$ ,  $(x, w)$  is in  $B_{\text{no}}$ .

Let  $p: \mathbb{Z}^+ \rightarrow \mathbb{N}$  be a non-decreasing polynomially bounded function, which will be fixed later. First notice that  $B$  has a qq-QAM proof system that satisfies completeness  $1 - 2^{-p}$  and soundness  $2^{-p}$  (the existence of such a proof system is ensured by Lemma 3.5). Starting from this qq-QAM proof system, the proof of Lemma 4.2 implies the existence of a polynomial-time algorithm that, given  $(x, w)$ , computes a description of a quantum circuit  $Q_{x,w}$  of  $q_{\text{in}}(|x|)$  input qubits and  $q_{\text{out}}(|x|)$  output qubits with the following properties:

- (i) if  $(x, w)$  is in  $B_{\text{yes}}$ , there exists a quantum state  $\rho$  consisting of  $q_{\text{in}}(|x|)$  qubits such that  $D(Q_{x,w}(\rho), (I/2)^{\otimes q_{\text{out}}(|x|)}) \leq 2^{-p(|x|+|w|)-1} < 2^{-p(|x|)}$ , and
- (ii) if  $(x, w)$  is in  $B_{\text{no}}$ , for any quantum state  $\rho$  consisting of  $q_{\text{in}}(|x|)$  qubits, it holds that  $D(Q_{x,w}(\rho), (I/2)^{\otimes q_{\text{out}}(|x|)}) > 1/20$ .

Let  $q: \mathbb{Z}^+ \rightarrow \mathbb{N}$  be another non-decreasing polynomially bounded function satisfying  $q(n) \geq \max\{l(n) + 4, n\}$  for any  $n$  in  $\mathbb{Z}^+$ . Considering the quantum circuit  $Q'_{x,w}$  that runs  $k(|x|)$  copies of  $Q_{x,w}$  in parallel for the polynomially bounded function  $k = \lceil \frac{2 \ln 2}{\ln(400/399)} q \rceil$  and taking  $p = q + \lceil \log k \rceil$ , it follows from Lemma 2.1 (with  $\Phi$  being the transformation induced by  $Q_{x,w}$  and  $\Psi$  being the transformation that receives an input state of  $q_{\text{in}}(|x|)$  qubits and always outputs the totally mixed state  $(I/2)^{\otimes q_{\text{out}}(|x|)}$  regardless of the input) that

- (i) if  $x$  is in  $A_{\text{yes}}$ , there exist a string  $w$  in  $\Sigma^{l(|x|)}$  and a quantum state  $\rho'$  consisting of  $q'_{\text{in}}(|x|)$  qubits such that  $D(Q'_{x,w}(\rho'), (I/2)^{\otimes q'_{\text{out}}(|x|)}) < 2^{-q(|x|)}$ , and
- (ii) if  $x$  is in  $A_{\text{no}}$ , for any string  $w$  in  $\Sigma^{l(|x|)}$  and any quantum state  $\rho'$  consisting of  $q'_{\text{in}}(|x|)$  qubits, it holds that  $D(Q'_{x,w}(\rho'), (I/2)^{\otimes q'_{\text{out}}(|x|)}) > 1 - 2^{-q(|x|)}$ ,

where  $q'_{\text{in}} = kq_{\text{in}}$  and  $q'_{\text{out}} = kq_{\text{out}}$ .

Now consider the quantum circuit  $R_x$  of  $(l(|x|) + q'_{\text{in}}(|x|))$  input qubits and  $q'_{\text{out}}(|x|)$  output qubits that corresponds to the following algorithm:

1. Measure all the  $l(|x|)$  qubits in the quantum register  $W$  in the computational basis to obtain a classical string  $w$  in  $\Sigma^{l(|x|)}$ , where  $W$  corresponds to the first  $l(|x|)$  qubits of the input qubits.
2. Compute from  $(x, w)$  a description of the quantum circuit  $Q'_{x,w}$ . Perform the circuit  $Q'_{x,w}$  with qubits in the quantum register  $R$  as its input qubits, where  $R$  corresponds to the last  $q'_{\text{in}}(|x|)$  qubits of the input qubits of  $R_x$ . Output the qubits corresponding to the output qubits of  $Q'_{x,w}$ .

We claim that the circuit  $R_x$  satisfies the following two properties:

- (i) if  $x$  is in  $A_{\text{yes}}$ , there exists a quantum state  $\sigma$  consisting of  $(l(|x|) + q'_{\text{in}}(|x|))$  qubits such that  $D(R_x(\sigma), (I/2)^{\otimes q'_{\text{out}}(|x|)}) < 2^{-q(|x|)}$ , and
- (ii) if  $x$  is in  $A_{\text{no}}$ , for any quantum state  $\sigma$  consisting of  $(l(|x|) + q'_{\text{in}}(|x|))$  qubits, it holds that  $D(R_x(\sigma), (I/2)^{\otimes q'_{\text{out}}(|x|)}) > 1/q'_{\text{out}}(|x|)$ .

In fact, the item (i) is obvious from the construction of  $R_x$ .

To prove the item (ii), suppose that  $x$  is in  $A_{\text{no}}$ . Then, for any string  $w$  in  $\Sigma^{l(|x|)}$  and any quantum state  $\rho'$  of  $q'_{\text{in}}(|x|)$  qubits, it holds that  $D(Q'_{x,w}(\rho'), (I/2)^{\otimes q'_{\text{out}}(|x|)}) > 1 - 2^{-q(|x|)}$ .

From Lemma 2.2 and the second inequality of Lemma 2.3, it follows that

$$S(R_x(\sigma)) < l(|x|) + q'_{\text{out}}(|x|) - q(|x|) + 2 \leq q'_{\text{out}}(|x|) - 2 \leq \left(1 - \frac{1}{q'_{\text{out}}(|x|)} - 2^{-q'_{\text{out}}(|x|)}\right) q'_{\text{out}}(|x|).$$

Hence, the first inequality of Lemma 2.3 ensures that  $D(R_x(\sigma), (I/2)^{\otimes q'_{\text{out}}(|x|)}) > 1/q'_{\text{out}}(|x|)$ .

Finally, consider the quantum circuit  $R'_x$  that runs  $k'(|x|)$  copies of  $R_x$  in parallel for the polynomially bounded function  $k' = \lceil \frac{2 \ln(1/2)}{\ln(1 - (1/q'_{\text{out}})^2)} \rceil \leq 2(q'_{\text{out}})^2$ . Assuming that  $(q'_{\text{out}}(|x|))^2 \leq 2^{q(|x|)-4}$  (otherwise  $|x|$  is at most some fixed constant since  $q'_{\text{out}}$  is a polynomially bounded function and  $q(|x|) \geq |x|$ , and thus, it can be checked trivially whether  $x$  is in  $A_{\text{yes}}$  or in  $A_{\text{no}}$ ), it follows from Lemma 2.1 that

- (i) if  $x$  is in  $A_{\text{yes}}$ , there exists a quantum state  $\sigma$  consisting of  $q''_{\text{in}}(|x|)$  qubits such that  $D(R'_x(\sigma), (I/2)^{\otimes q'_{\text{out}}(|x|)}) < 1/8$ , and
- (ii) if  $x$  is in  $A_{\text{no}}$ , for any quantum state  $\sigma$  consisting of  $q''_{\text{in}}(|x|)$  qubits, it holds that  $D(R'_x(\sigma), (I/2)^{\otimes q'_{\text{out}}(|x|)}) > 1/2$ ,

where  $q''_{\text{in}} = k'(l + q'_{\text{in}})$  and  $q'_{\text{out}} = k'q'_{\text{out}}$ .

Thus,  $R'_x$  is a yes-instance of CITM(1/8, 1/2) if  $x$  is in  $A_{\text{yes}}$ , while  $R'_x$  is a no-instance of CITM(1/8, 1/2) if  $x$  is in  $A_{\text{no}}$ . This implies that any problem  $A$  in ccq-QAM is reducible to CITM(1/8, 1/2) in polynomial time, and thus in qq-QAM by Lemma 4.1, which completes the proof.  $\blacktriangleleft$

► **Remark.** Combined with Lemma 2.3, the reduction from the problem  $B$  to the circuit  $Q'_{x,w}$  in the proof of Lemma 5.2 essentially shows the qq-QAM-hardness of the MAXOUTQEA problem. On the other hand, the fact that MAXOUTQEA is in qq-QAM is easily proved by a straightforward modification of the arguments in Refs. [6, 8] that place the QUANTUM ENTROPY APPROXIMATION (QEA) problem in NIQSZK. Hence, the MAXOUTQEA problem is also qq-QAM-complete, giving Theorem 1.3. A rigorous proof of MAXOUTQEA being in qq-QAM is presented in Appendix A, and a separate proof of the qq-QAM-hardness of MAXOUTQEA is found in the full version [19] of this paper.

Finally, using Lemma 5.2, it is proved that ccq-QAM  $\subseteq$  qq-QAM.

► **Lemma 5.3.** ccq-QAM  $\subseteq$  qq-QAM.

**Proof.** Let  $A = (A_{\text{yes}}, A_{\text{no}})$  be a problem in ccq-QAM. By Lemma 3.5, one can assume that  $A$  has a ccq-QAM proof system  $\Pi$  with completeness  $1 - 2^{-8}$  and soundness  $2^{-8}$ . Let  $l: \mathbb{Z}^+ \rightarrow \mathbb{N}$  be the polynomially bounded function that specifies the length of the first message in  $\Pi$ . Consider the ccq-QAM proof system  $\Pi^{(-1)}$  that on input  $(x, r)$  simulates the last three turns of  $\Pi$  on  $x$  assuming that the first message in  $\Pi$  was  $r$  in  $\Sigma^{l(|x|)}$ . Let  $B = (B_{\text{yes}}, B_{\text{no}})$  be the following promise problem in ccq-QAM:

$$B_{\text{yes}} = \{(x, r): \text{MAP}_{(x,r)}(\Pi^{(-1)}) \geq 2/3\}, \quad B_{\text{no}} = \{(x, r): \text{MAP}_{(x,r)}(\Pi^{(-1)}) \leq 1/3\}.$$

Note that, if  $x$  is in  $A_{\text{yes}}$ , then  $(x, r)$  is in  $B_{\text{yes}}$  for at least  $(1 - 3 \cdot 2^{-8})$ -fraction of the choices of  $r$ , while if  $x$  is in  $A_{\text{no}}$ , then  $(x, r)$  is in  $B_{\text{no}}$  for at least  $(1 - 3 \cdot 2^{-8})$ -fraction of the choices of  $r$ . By Lemma 5.2, it holds that  $B$  is in qq-QAM. Thus, there exists a qq-QAM proof system  $\Pi'$  for  $B$  such that  $\text{MAP}_{(x,r)}(\Pi')$  is at least  $2/3$  if  $(x, r)$  is in  $B_{\text{yes}}$ , while  $\text{MAP}_{(x,r)}(\Pi')$  is at most  $1/3$  if  $(x, r)$  is in  $B_{\text{no}}$ . Here, the first turn of  $\Pi'$  is a turn for the verifier, and thus one can merge the turn for sending  $r$  with the first turn of  $\Pi'$ . This results in another qq-QAM proof system  $\Pi''$  for  $A$  in which at the first turn the new verifier sends a string  $r$  in  $\Sigma^{l(|x|)}$  chosen uniformly at random in addition to the original first message of the verifier



in  $\Pi'$  on input  $(x, r)$ , and then behaves exactly in the same manner as the verifier in  $\Pi'$  on input  $(x, r)$  in the rest of the protocol. Notice that sending a random string  $r$  of length  $l(|x|)$  can be exactly simulated by sending the halves of  $l(|x|)$  EPR pairs and measuring in the computational basis all the remaining halves of them that the verifier possesses. If  $x$  is in  $A_{\text{yes}}$ ,  $\text{MAP}_x(\Pi'')$  is at least  $(1 - 3 \cdot 2^{-8}) \cdot (2/3) > 5/8$ , while if  $x$  is in  $A_{\text{no}}$ ,  $\text{MAP}_x(\Pi'')$  is at most  $3 \cdot 2^{-8} + (1 - 3 \cdot 2^{-8}) \cdot (1/3) < 3/8$ , which is sufficient for the claim, due to Lemma 3.5. ◀

Now one inclusion of Theorem 1.1 is immediate from Lemmas 5.1 and 5.3, and the other inclusion is trivial, which completes the proof of Theorem 1.1.

In fact, all the proofs of Lemmas 5.1, 5.2, and 5.3 can be easily modified to preserve the perfect completeness property, and the following corollary holds.

► **Corollary 5.4.** *For any constant  $m \geq 2$ ,  $c \cdots \text{cqq-QAM}_1(m) = \text{qq-QAM}_1$ .*

**Proof.** The proof of Lemma 5.1 can be modified so that it preserves the perfect completeness property by taking  $B_{\text{yes}}$  to be the set of  $(x, r)$ 's such that  $\text{MAP}_{(x,r)}(\Pi^{(-1)})$  is one, and using Lemma 3.6 instead of Lemma 3.5. This shows that  $c \cdots \text{cqq-QAM}_1(m)$  is included in  $\text{ccqq-QAM}_1$  for any constant  $m \geq 4$ . With a similar modification to the set  $B_{\text{yes}}$  as well as using Corollary 4.3 instead of Theorem 1.2, the proof of Lemma 5.2 can be modified to present a reduction from any problem in  $\text{cqq-QAM}_1$  to  $\text{CITM}(0, b)$ , which shows that  $\text{cqq-QAM}_1$  is included in  $\text{qq-QAM}_1$ . Using this inclusion instead of Lemma 5.2 and again with a similar modification to  $B_{\text{yes}}$  and a replacement of Lemma 3.5 by Lemma 3.6, the proof of Lemma 5.3 can be modified so that  $\text{ccqq-QAM}_1$  is shown to be in  $\text{qq-QAM}_1$ . ◀

## 6 QAM versus one-sided error qq-QAM

This section shows that qq-QAM proof systems of perfect-completeness are already as powerful as the standard QAM proof systems of two-sided bounded error (Theorem 1.4). As mentioned at the end of Section 5, the collapse theorem for qq-QAM holds even for the perfect-completeness variants. In particular, the inclusion  $\text{ccqq-QAM}_1 \subseteq \text{qq-QAM}_1$  holds. Hence, for the proof of Theorem 1.4, it suffices to show that any problem in  $\text{cq-QAM}$  (= QAM) is necessarily in the class  $\text{ccqq-QAM}_1$ . As mentioned earlier, this can be shown by combining the classical technique in Ref. [7] for proving  $\text{AM} = \text{AM}_1$ , which originates in the proof of  $\text{BPP} \subseteq \Sigma_2^{\text{P}}$  due to Lautemann [22], and the recent result that sharing a constant number of EPR pairs can make QMA proofs perfectly complete [20].

**Proof of Theorem 1.4 (Sketch).** Intuitively, with two classical turns of communications, the classical technique in Ref. [7] can be used to generate polynomially many instances of a (promise) QMA problem such that all these instances are QMA yes-instances if the input was a yes-instance, while at least one of these instances is a QMA no-instance with high probability if the input was a no-instance (some of the QMA instances may violate the promise if the input was a no-instance, but this does not matter, as the important point is that at least one instance is a no-instance in this case). Now one makes use of the  $\text{QMA}_1^{\text{const-EPR}}$  proof system in Ref. [20] for each QMA instance, by running polynomially many attempts of such a system in parallel to see that none of them results in rejection. The resulting proof system is thus of  $\text{ccqq-QAM}$  type, as  $\text{QMA}_1^{\text{const-EPR}}$  proof systems are special cases of qq-QAM proof systems. The perfect completeness of this proof system follows from the fact that all the QMA instances generated from an input of yes-instance are QMA yes-instances, and all of them are accepted without error in the attempts of the  $\text{QMA}_1^{\text{const-EPR}}$  system due to the perfect completeness property of the system. The soundness of this proof



system follows from the fact that at least one QMA instance generated from an input of no-instance is a QMA no-instance with high probability, for which the  $\text{QMA}_1^{\text{const-EPR}}$  proof system results in rejection with reasonably high probability, due to the soundness property of it. The rigorous proof is found in the full version [19] of this paper. ◀

The fact that perfect completeness is achievable in cc-QAM (Theorem 1.6) can be proved in a similar fashion, except that now one uses the fact  $\text{MQA} = \text{MQA}_1$  (a.k.a.,  $\text{QCMA} = \text{QCMA}_1$ ) that any classical-witness QMA proofs can be made perfectly complete shown in Ref. [15] instead of the inclusion  $\text{QMA} \subseteq \text{QMA}_1^{\text{const-EPR}}$ . Each QMA instance in the argument above are replaced by an MQA (QCMA) instance in this case. Notice that no additional turn is necessary in this case, as the second turn is a classical turn for a prover and witnesses for the MQA instances can be sent also at this turn. Hence, the resulting proof system corresponding to  $\Pi''$  is immediately a cc-QAM proof system of perfect completeness.

## 7 Collapse theorem for general quantum Arthur-Merlin proof systems

Before the proof of Theorem 1.7, first observe the simple fact that one can always replace classical turns by quantum ones without diminishing the verification power, by letting the verifier simulate classical turns by quantum turns via CNOT applications.

► **Proposition 7.1.** *For any constant  $m$  in  $\mathbb{N}$ , any  $j$  in  $\{1, \dots, m\}$ , and any message-types  $t_1, \dots, t_m$  in  $\{c, q\}$ ,*

$$t_m \cdots t_{j+1} t_j t_{j-1} \cdots t_1\text{-QAM}(m) \subseteq t_m \cdots t_{j+1} q t_{j-1} \cdots t_1\text{-QAM}(m).$$

As generalized quantum Arthur-Merlin proofs are nothing but a special case of general quantum interactive proofs, it is obvious that for any constant  $m$  and any message-types  $t_1, \dots, t_m$  in  $\{c, q\}$ ,  $t_m \cdots t_1\text{-QAM}(m)$  is contained in  $\text{QIP} = \text{PSPACE}$  [13]. As mentioned in Section 1, Marriott and Watrous [24] proved that qcq-QAM (= QMAM) already hits the ceiling, i.e., coincides with QIP. Next lemma states that one can slightly improve this and even the third message is not necessary to be quantum to have the full power of quantum interactive proofs. The proof is based on a simulation of the original qcq-QAM system by a qcc-QAM system using quantum teleportation.

► **Lemma 7.2.**  $\text{qcq-QAM} \subseteq \text{qcc-QAM}$ .

**Proof (Sketch).** Let  $A = (A_{\text{yes}}, A_{\text{no}})$  be a problem in qcq-QAM, meaning that  $A$  has a qcq-QAM proof system  $\Pi$  with completeness  $2/3$  and soundness  $1/3$  that is specified by the protocol of the verifier of the following form for every input  $x$ :

1. Receive a quantum register  $M_1$  from the prover, and then send a random string  $r$  to the prover.
2. Receive a quantum register  $M_2$  from the prover. Prepare a private quantum register  $V$ , and perform the final verification procedure over  $(M_1, M_2, V)$ .

Let  $l: \mathbb{Z}^+ \rightarrow \mathbb{N}$  be the polynomially bounded function that specifies the number of qubits in  $M_2$ . Consider the teleportation-based simulation of  $\Pi$  by the qcc-QAM proof system  $\tilde{\Pi}$ , where the verifier performs the following protocol for every input  $x$ :

1. Receive a quantum register  $S_1$  of  $l(|x|)$  qubits, in addition to the quantum register  $M_1$ , from the prover. Send a random string  $r$  to the prover as would be done in  $\Pi$ .

2. Receive a binary string  $b$  of length  $2l(|x|)$  from the prover. Apply  $X^{b_{j,1}}Z^{b_{j,2}}$  to the  $j$ th qubit of  $S_1$ , for each  $j$  in  $\{1, \dots, l(|x|)\}$ , where  $b_{j,1}$  and  $b_{j,2}$  denote the  $(2j-1)$ st and  $(2j)$ th bits of  $b$ , respectively. Finally, prepare his/her private quantum register  $V$  as in  $\Pi$ , and simulate the final verification procedure of the verifier in  $\Pi$  with  $(M_1, S_1, V)$ .

The claim follows from a rigorous analysis of this protocol, which is relegated to the full version [19] of this paper. ◀

With Lemma 7.2 in hand, Theorem 1.7 is proved as follows.

**Proof of Theorem 1.7.** For the item (i), first notice that qcq-QAM is shown to be in qccc-QAM by an argument very similar to the proof of Lemma 7.2, with not the honest prover but the verifier preparing the EPR pairs. As qcq-QAM = QMAM = QIP = PSPACE, together with Lemma 7.2, this implies that qccc-QAM = qcc-QAM = PSPACE. As adding more turns to  $qt_3t_2t_1$ -QAM and  $qt_2t_1$ -QAM proof systems does not diminish the verification power for any  $t_1, t_2$ , and  $t_3$  in  $\{q, c\}$ , this establishes the claim in the item (i).

For the item (ii), again with a similar argument to the proof of Lemma 7.2, it holds that  $c \cdots \text{cq}q\text{-QAM}(m)$  is included in  $c \cdots \text{qc}q\text{-QAM}(m)$  for any constant  $m \geq 2$ , and thus, combined with Theorem 1.1 and Proposition 7.1, the claim follows.

For the item (iii), it suffices to show that, for any constant  $m \geq 3$ ,  $c \cdots \text{c}q\text{-QAM}(m)$  is included in  $c \cdots \text{c}q\text{-QAM}(m-1)$ . The case with  $m \geq 5$  is proved with an argument similar to that in the proof of Lemma 5.1, since the first three (resp. four) turns of the  $m$ -turn  $c \cdots \text{c}q\text{-QAM}$  proof systems are classical when  $m$  is odd (resp. when  $m$  is even). In the case where  $m = 3$ , one modifies the construction of  $\Pi'$  in the proof of Lemma 5.1 so that the message from the prover at the second turn (corresponding to Step 2 of  $\Pi'$ ) is quantum, consisting of  $(k(|x|) + 1)$  parts: the  $Y$  part and each  $Z_j$  part for  $j$  in  $\{1, \dots, k(|x|)\}$ , corresponding to  $y$  and each  $z_j$  in Step 2 of  $\Pi'$ . In order to force the content in the  $Y$  part to be classical, the verifier simply measures each qubit in the  $Y$  part in the computational basis. The analysis in the proof of Lemma 5.1 then works with the case where  $m = 3$ , i.e., the case where a  $\text{cc}q\text{-QAM}$  system is simulated by a  $\text{c}q\text{-QAM}$  system. The case where  $m = 4$  can then be proved using this result with  $m = 3$ , with the same argument as in the proof of Lemma 5.1.

Finally, for the item (iv), it suffices to show that, for any constant  $m \geq 3$ ,  $c \cdots \text{c}\text{-QAM}(m)$  is included in  $c \cdots \text{c}\text{-QAM}(m-1)$ , which easily follows from an argument similar to that in the proof of Lemma 5.1, since all the messages are classical. ◀

## 8 Conclusion

This paper has introduced the generalized model of quantum Arthur-Merlin proof systems to provide some new insights on the power of two-turn quantum interactive proofs. A number of open problems are listed below concerning generalized quantum Arthur-Merlin proof systems and other related topics:

- Is there any natural problem, other than CITM and MAXOUTQEA, in  $\text{qq}\text{-QAM}$  that is not known to be in the standard QAM? Or is  $\text{qq}\text{-QAM}$  equal to QAM?
- Currently no upper-bound is known for  $\text{qq}\text{-QAM}$  other than QIP(2). Can a better upper-bound be placed on  $\text{qq}\text{-QAM}$ ? Is  $\text{qq}\text{-QAM}$  contained in  $\text{BP} \cdot \text{PP}$ ?
- Does  $\text{qq}\text{-QAM} = \text{qq}\text{-QAM}_1$ ? In other words, is perfect completeness achievable in  $\text{qq}\text{-QAM}$ ? Similar questions remain open even for QIP(2) and QAM.
- What happens if some of the messages are restricted to be classical in the standard quantum interactive proof systems? Does a collapse theorem similar to the  $\text{qq}\text{-QAM}$  case hold even with the QIP(2) case? More precisely, is the power of  $m$ -turn quantum

interactive proof systems equivalent to QIP(2) for any constant  $m \geq 2$ , when the first  $(m - 2)$  turns are restricted to exchange only classical messages?

For the last question above, note that one might be able to show a similar collapse theorem even with QIP(2) when the verifier *cannot* use quantum operations at all during the first  $(m - 2)$  turns (possibly by extending the argument due to Goldwasser and Sipser [10] to replace the classical interaction of the first  $(m - 2)$  turns by an  $m$ -turn classical public-coin interaction, and then applying arguments similar to those in this paper, using some appropriate QIP(2)-complete problem like the CLOSE IMAGE problem [32, 12], although the authors do not know if this approach works). A more difficult, but more natural and interesting case is where the verifier can use quantum operations to generate his/her classical messages even for the first  $(m - 2)$  turns, to which the Goldwasser-Sipser technique does not seem to apply any longer. A collapse theorem for such a case, if provable, would be very helpful when trying to put more problems in QIP(2) and more generally investigating the properties of two-turn quantum interactive proof systems.

**Acknowledgements.** The authors are grateful to Francesco Buscemi and Richard Cleve for very useful discussions. This work is supported by the Grant-in-Aid for Scientific Research (A) No. 24240001 of the Japan Society for the Promotion of Science and the Grant-in-Aid for Scientific Research on Innovative Areas No. 24106009 of the Ministry of Education, Culture, Sports, Science and Technology in Japan. HN also acknowledges support from the Grant-in-Aids for Scientific Research (A) Nos. 21244007 and 23246071 and (C) No. 25330012 of the Japan Society for the Promotion of Science.

---

## References

- 1 Scott Aaronson, Salman Beigi, Andrew Drucker, Bill Fefferman, and Peter Shor. The power of unentanglement. *Theory of Computing*, 5:1–42 (article 1), 2009.
- 2 Dorit Aharonov. A simple proof that Toffoli and Hadamard are quantum universal. arXiv.org e-Print archive, arXiv:quant-ph/0301040, 2003.
- 3 Dorit Aharonov, Alexei Kitaev, and Noam Nisan. Quantum circuits with mixed states. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, pages 20–30, 1998.
- 4 László Babai. Trading group theory for randomness. In *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing*, pages 421–429, 1985.
- 5 Salman Beigi, Peter Shor, and John Watrous. Quantum interactive proofs with short messages. *Theory of Computing*, 7:101–117 (article 7), 2011.
- 6 Avraham Ben-Aroya, Oded Schwartz, and Amnon Ta-Shma. Quantum expanders: Motivation and constructions. *Theory of Computing*, 6:47–79 (article 3), 2010.
- 7 Jin-Yi Cai. Lectures in computational complexity, August 2012. Available at <http://www.cs.wisc.edu/~jyc/710/book.pdf>.
- 8 André Chailloux, Dragos Florin Ciocan, Iordanis Kerenidis, and Salil Vadhan. Interactive and noninteractive zero knowledge are equivalent in the help model. In *Theory of Cryptography, Fifth Theory of Cryptography Conference, TCC 2008*, volume 4948 of *Lecture Notes in Computer Science*, pages 501–534, 2008. A full version available as Cryptology ePrint Archive, Report 2007/467, 2007.
- 9 Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208, 1989.
- 10 Shafi Goldwasser and Michael Sipser. Private coins versus public coins in interactive proof systems. In Silvio Micali, editor, *Randomness and Computation*, volume 5 of *Advances in Computing Research*, pages 73–90. JAI Press, 1989.

- 11 Gustav Gutoski. *Quantum Strategies and Local Operations*. PhD thesis, David R. Cheriton School of Computer Science, University of Waterloo, 2009. arXiv.org e-Print archive, arXiv:1003.0038 [quant-ph].
- 12 Patrick Hayden, Kevin Milner, and Mark M. Wilde. Two-message quantum interactive proofs and the quantum separability problem. *Quantum Information and Computation*, 14(5–6):0384–0416, 2014.
- 13 Rahul Jain, Zhengfeng Ji, Sarvagya Upadhyay, and John Watrous. QIP = PSPACE. *Journal of the ACM*, 58(6):article 30, 2011.
- 14 Rahul Jain, Sarvagya Upadhyay, and John Watrous. Two-message quantum interactive proofs are in PSPACE. In *50th Annual Symposium on Foundations of Computer Science*, pages 534–543, 2009.
- 15 Stephen P. Jordan, Hirotada Kobayashi, Daniel Nagaj, and Harumichi Nishimura. Achieving perfect completeness in classical-witness quantum Merlin-Arthur proof systems. *Quantum Information and Computation*, 12(5–6):0461–0471, 2012.
- 16 Alexei Kitaev and John Watrous. Parallelization, amplification, and exponential time simulation of quantum interactive proof systems. In *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing*, pages 608–617, 2000.
- 17 Alexei Yu. Kitaev, Alexander H. Shen, and Mikhail N. Vyalı. *Classical and Quantum Computation*, volume 47 of *Graduate Studies in Mathematics*. American Mathematical Society, 2002.
- 18 Hirotada Kobayashi. Non-interactive quantum perfect and statistical zero-knowledge. In *Algorithms and Computation, 14th International Symposium, ISAAC 2003*, volume 2906 of *Lecture Notes in Computer Science*, pages 178–188, 2003.
- 19 Hirotada Kobayashi, François Le Gall, and Harumichi Nishimura. Generalized quantum Arthur-Merlin games. arXiv.org e-Print archive, arXiv:1312.4673v2 [quant-ph], 2014.
- 20 Hirotada Kobayashi, François Le Gall, and Harumichi Nishimura. Stronger methods of making quantum interactive proofs perfectly complete. *SIAM Journal on Computing*, 44(2):243–289, 2015.
- 21 Hirotada Kobayashi, Keiji Matsumoto, and Tomoyuki Yamakami. Quantum Merlin-Arthur proof systems: Are multiple Merlins more helpful to Arthur? *Chicago Journal of Theoretical Computer Science*, 2009:article 3, 2009.
- 22 Clemens Lautemann. BPP and the polynomial hierarchy. *Information Processing Letters*, 17(4):215–217, 1983.
- 23 Carsten Lund, Lance Fortnow, Howard Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. *Journal of the ACM*, 39(4):859–868, 1992.
- 24 Chris Marriott and John Watrous. Quantum Arthur-Merlin games. *Computational Complexity*, 14(2):122–152, 2005.
- 25 Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.
- 26 Christos H. Papadimitriou. Games against nature. *Journal of Computer and System Sciences*, 31(2):288–301, 1985.
- 27 Bill Rosgen and John Watrous. On the hardness of distinguishing mixed-state quantum computations. In *Twentieth Annual IEEE Conference on Computational Complexity*, pages 344–354, 2005.
- 28 Adi Shamir. IP = PSPACE. *Journal of the ACM*, 39(4):869–877, 1992.
- 29 Alexander Shen. IP = PSPACE: Simplified proof. *Journal of the ACM*, 39(4):878–880, 1992.
- 30 Yaoyun Shi. Both Toffoli and Controlled-NOT need little help to do universal quantum computing. *Quantum Information and Computation*, 3(1):084–092, 2003.

- 31 Peter W. Shor. Fault-tolerant quantum computation. In *37th Annual Symposium on Foundations of Computer Science*, pages 56–65, 1996.
- 32 John Watrous. Capturing quantum complexity classes via quantum channels. Talk at the 6th Workshop on Quantum Information Processing, December 2002.
- 33 John Watrous. Limits on the power of quantum statistical zero-knowledge. In *43rd Annual Symposium on Foundations of Computer Science*, pages 459–468, 2002.
- 34 John Watrous. PSPACE has constant-round quantum interactive proof systems. *Theoretical Computer Science*, 292(3):575–588, 2003.
- 35 John Watrous. Zero-knowledge against quantum attacks. *SIAM Journal on Computing*, 39(1):25–58, 2009.
- 36 Stephanie Wehner. Entanglement in interactive proof systems with binary answers. In *STACS 2006, 23rd Annual Symposium on Theoretical Aspects of Computer Science*, volume 3884 of *Lecture Notes in Computer Science*, pages 162–171, 2006.
- 37 Mark M. Wilde. *Quantum Information Theory*. Cambridge University Press, 2013.

## A

 qq-QAM-completeness of MaxOutQEA

This section presents a proof of the MAXOUTQEA problem being in qq-QAM. As the proof of Lemma 5.2 essentially shows the qq-QAM-hardness of MAXOUTQEA (a separate proof of which is found in the full version [19] of this paper), this proves Theorem 1.3, the qq-QAM-completeness of MAXOUTQEA.

► **Lemma A.1.** MAXOUTQEA is in qq-QAM.

**Proof.** We present a reduction from the MAXOUTQEA problem to the CITM problem (with some appropriate parameters), by modifying the reduction from the QEA problem to the QUANTUM STATE CLOSENESS TO TOTALLY MIXED (QSCTM) problem presented in the full version of Ref. [8], which relies on the analysis found in Section 5.3 of Ref. [6].

Let  $x = (Q, t)$  be an instance of MAXOUTQEA, where  $Q$  is a description of a quantum circuit that specifies a quantum channel  $\Phi$ , and  $t$  is a positive integer. For simplicity, in what follows, we identify the description  $Q$  and the quantum circuit it induces. Suppose that  $Q$  acts on  $m_{\text{all}}$  qubits with  $m_{\text{in}}$  specified input qubits and  $m_{\text{out}}$  specified output qubits. Let  $q$  and  $\varepsilon$  be two functions that appear in Eqs. (5.1) and (5.2) of Ref. [6]<sup>1</sup> to be specified later. We consider the quantum circuit  $Q^{\otimes q(|x|)}$  that runs  $q(|x|)$  copies of  $Q$  in parallel, and the  $(qt, d, \varepsilon)$ -quantum extractor  $E$  on  $q(|x|) m_{\text{out}}$  qubits given in Ref. [6, Section 5.3], which is written as  $E = \frac{1}{2^d} \sum_{j=1}^{2^d} E_j$ , where  $E_j(\rho) = U_j \rho U_j^\dagger$  for unitary operators  $U_j$ . Let  $R$  be the quantum circuit that runs  $Q^{\otimes q(|x|)}$  and then applies  $E$  to the output state of  $q(|x|) m_{\text{out}}$  qubits. By following the analysis found in Ref. [6], one can show that

- (i) if  $x = (Q, t)$  is a yes-instance of MAXOUTQEA, there exists a quantum state  $\rho$  consisting of  $q(|x|) m_{\text{in}}$  qubits such that  $D(R(\rho), (I/2)^{\otimes q(|x|) m_{\text{out}}}) \leq \frac{3}{2}\varepsilon$ , and
- (ii) if  $x = (Q, t)$  is a no-instance of MAXOUTQEA, for any quantum state  $\rho$  consisting of  $q(|x|) m_{\text{in}}$  qubits, it holds that  $D(R(\rho), (I/2)^{\otimes q(|x|) m_{\text{out}}}) \geq \frac{1}{4q(|x|) m_{\text{out}}}$ .

In fact, the item (i) follows from exactly the same analysis as in Ref. [6], by taking  $\rho = \sigma^{\otimes q(|x|)}$  with  $\sigma$  being a quantum state of  $m_{\text{in}}$  qubits such that  $S(Q(\sigma)) \geq t + 1$  (the condition  $S_{\text{max}}(\Phi) \geq t + 1$  ensures the existence of such a state  $\sigma$ ).

<sup>1</sup> Rigorously speaking,  $q$  in the present case corresponds to  $\frac{q}{2}$  in the left-hand sides of Eqs. (5.1) and (5.2) of Ref. [6]. This is due to the fact that the MAXOUTQEA problem in this paper is defined using threshold values  $t + 1$  and  $t - 1$ , while the QEA problem in Ref. [6] is defined using threshold values  $t + \frac{1}{2}$  and  $t - \frac{1}{2}$ .

To prove the item (ii), first notice that, if  $x = (Q, t)$  is a no-instance of MAXOUTQEA, it holds that  $S(Q(\sigma)) \leq S_{\max}(\Phi) \leq t - 1$  for any quantum state  $\sigma$  of  $m_{\text{in}}$  qubits. Take an arbitrary quantum state  $\rho$  of  $q(|x|) m_{\text{in}}$  qubits. By Lemma 2.2, it holds that

$$S(R(\rho)) = S\left(\frac{1}{2^d} \sum_{j=1}^{2^d} U_j Q^{\otimes q(|x|)}(\rho) U_j^\dagger\right) \leq S(Q^{\otimes q(|x|)}(\rho)) + d.$$

For each  $j$  in  $\{1, \dots, q(|x|)\}$ , let  $R_j$  be the output quantum register of the  $j$ th copy of  $Q$  (hence, the whole output state  $Q^{\otimes q(|x|)}(\rho)$  of  $Q^{\otimes q(|x|)}$  is in  $(R_1, \dots, R_{q(|x|)})$ ), and let  $\sigma_{R_j}$  be the reduced state of  $Q^{\otimes q(|x|)}(\rho)$  of  $m_{\text{out}}$  qubits obtained by tracing out all the qubits except those in  $R_j$ . By the subadditivity of von Neumann entropy, it follows that

$$S(Q^{\otimes q(|x|)}(\rho)) \leq \sum_{j=1}^{q(|x|)} S(\sigma_{R_j}) \leq \sum_{j=1}^{q(|x|)} \max_{\sigma} S(Q(\sigma)) \leq (t - 1) q(|x|),$$

which implies that

$$S(R(\rho)) \leq (t - 1) q(|x|) + d.$$

Now the item (ii) follows from exactly the same analysis as in Ref. [6].

To complete the reduction, similarly to the full version of Ref. [8], one takes  $\varepsilon = 2^{-k}$  for a polynomially bounded function  $k: \mathbb{Z}^+ \rightarrow \mathbb{N}$  such that  $k(n) \geq n$  for any  $n$  in  $\mathbb{Z}^+$  and  $k(n) \in O(n)$ , and a polynomially bounded function  $q: \mathbb{Z}^+ \rightarrow \mathbb{N}$  such that  $q(n) \in \Theta(n^4)$  so that Eqs. (5.1) and (5.2) of Ref. [6] are satisfied. Consider the quantum circuit  $R'$  that runs  $r(|x|)$  copies of  $R$  in parallel for a polynomially bounded function  $r: \mathbb{Z}^+ \rightarrow \mathbb{N}$  such that  $r(n) = \lceil \frac{2 \ln(1/2)}{\ln(1 - (1/(2q(n)m_{\text{out}})^2))} \rceil \leq 2(2q(n)m_{\text{out}})^2$  for all  $n$  in  $\mathbb{Z}^+$ . Assuming that  $r(|x|) \leq 2^{|x|}/12$  (otherwise  $|x|$  is at most some fixed constant as  $r$  is a polynomially bounded function, and thus, it can be checked trivially whether  $x = (Q, t)$  is a yes-instance or a no-instance), it follows from Lemma 2.1 that

- (i) if  $x = (Q, t)$  is a yes-instance of MAXOUTQEA, there exists a quantum state  $\sigma$  consisting of  $r(|x|) q(|x|) m_{\text{in}}$  qubits such that  $D(R'(\sigma), (I/2)^{\otimes r(|x|) q(|x|) m_{\text{out}}}) \leq 1/8$ , and
- (ii) if  $x = (Q, t)$  is a no-instance of MAXOUTQEA, for any quantum state  $\sigma$  consisting of  $r(|x|) q(|x|) m_{\text{in}}$  qubits, it holds that  $D(R'(\sigma), (I/2)^{\otimes r(|x|) q(|x|) m_{\text{out}}}) \geq 1/2$ .

Hence, MAXOUTQEA is reducible to CITM(1/8, 1/2) in polynomial time, and thus in qq-QAM by Lemma 4.1. ◀



# Parallel Repetition for Entangled $k$ -player Games via Fast Quantum Search

Kai-Min Chung<sup>1</sup>, Xiaodi Wu<sup>2</sup>, and Henry Yuen<sup>3</sup>

- 1 Academia Sinica, Institute of Information Science  
128 Academia Road, Nankang, Taipei 11529, Taiwan  
kmchung@iis.sinica.edu.tw
- 2 MIT Center for Theoretical Physics  
77 Massachusetts Ave, Cambridge MA 02139, USA  
xiaodiwu@mit.edu
- 3 MIT CSAIL  
32 Vassar St, Cambridge MA 02139, USA  
hyuen@mit.edu

---

## Abstract

We present two parallel repetition theorems for the entangled value of multi-player, one-round free games (games where the inputs come from a product distribution). Our first theorem shows that for a  $k$ -player free game  $G$  with entangled value  $\text{val}^*(G) = 1 - \epsilon$ , the  $n$ -fold repetition of  $G$  has entangled value  $\text{val}^*(G^{\otimes n})$  at most  $(1 - \epsilon^{3/2})^{\Omega(n/sk^4)}$ , where  $s$  is the answer length of any player. In contrast, the best known parallel repetition theorem for the *classical* value of two-player free games is  $\text{val}(G^{\otimes n}) \leq (1 - \epsilon^2)^{\Omega(n/s)}$ , due to Barak, et al. (RANDOM 2009). This suggests the possibility of a separation between the behavior of entangled and classical free games under parallel repetition.

Our second theorem handles the broader class of free games  $G$  where the players can output (possibly entangled) quantum states. For such games, the repeated entangled value is upper bounded by  $(1 - \epsilon^2)^{\Omega(n/sk^2)}$ . We also show that the dependence of the exponent on  $k$  is necessary: we exhibit a  $k$ -player free game  $G$  and  $n \geq 1$  such that  $\text{val}^*(G^{\otimes n}) \geq \text{val}^*(G)^{n/k}$ .

Our analysis exploits the novel connection between communication protocols and quantum parallel repetition, first explored by Chailloux and Scarpa (ICALP 2014). We demonstrate that better communication protocols yield better parallel repetition theorems: in particular, our first theorem crucially uses a quantum search protocol by Aaronson and Ambainis, which gives a quadratic Grover speed-up for distributed search problems. Finally, our results apply to a broader class of games than were previously considered before; in particular, we obtain the first parallel repetition theorem for entangled games involving more than two players, and for games involving quantum outputs.

**1998 ACM Subject Classification** F.1.2. Modes of Computation

**Keywords and phrases** Parallel repetition, quantum entanglement, communication complexity

**Digital Object Identifier** 10.4230/LIPIcs.CCC.2015.512

## 1 Introduction

The study of multi-player one-round games has been central to both theoretical computer science and quantum information. Games have served as an indispensable tool with which to study a diverse array of topics, from the hardness of approximation to cryptography; from delegated computation to Bell inequalities; from proof systems to the monogamy of



© Kai-Min Chung, Xiaodi Wu, and Henry Yuen;  
licensed under Creative Commons License CC-BY  
30th Conference on Computational Complexity (CCC'15).

Editor: David Zuckerman; pp. 512–536



Leibniz International Proceedings in Informatics

LIPIcs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



entanglement. In particular, two-player games have received the most scrutiny. In a two-player game  $G$ , a referee samples a pair of questions  $(x, y)$  from some distribution  $\mu$ , and sends question  $x$  to one player (typically named Alice), and  $y$  to the other (typically named Bob). Alice and Bob then utilize some non-communicating strategy to produce answers  $a$  and  $b$ , respectively, upon which the referee computes some predicate  $V(x, y, a, b)$  to decide whether to accept or not. In this paper, we focus on the setting where Alice and Bob may utilize quantum entanglement as part of their strategy. The primary quantity of interest is the *entangled value*  $\text{val}^*(G)$  of game  $G$ , which is the maximum success probability over all possible entangled strategies for the players.

Recently, there has been significant interest in the *parallel repetition* of entangled games [15, 6, 7, 13, 9]. More formally, the  $n$ -fold parallel repetition of a game  $G$  is a game  $G^{\otimes n}$  where the referee will sample  $n$  independent pairs of questions  $(x_1, y_1), \dots, (x_n, y_n)$  from the distribution  $\mu$ . Alice receives  $(x_1, \dots, x_n)$  and Bob receives  $(y_1, \dots, y_n)$ . They produce outputs  $(a_1, \dots, a_n)$  and  $(b_1, \dots, b_n)$ , respectively, and they win only if  $V(x_i, y_i, a_i, b_i) = 1$  for all  $i$ . We call each  $i$  a “coordinate” of  $G^{\otimes n}$  or “repetition” of  $G$ .

Suppose we have a game  $G$  where  $\text{val}^*(G) = 1 - \epsilon$ . Intuitively, one should expect that  $\text{val}^*(G^{\otimes n})$  should behave as  $(1 - \epsilon)^n$ . Indeed, this would be the case if the game  $G$  were played  $n$  times *sequentially*. However, there are counterexamples of games  $G$  and  $n > 1$  where  $\text{val}^*(G^{\otimes n}) = \text{val}^*(G)$  (e.g., as in [8]). Despite such counterexamples, it has been shown that the *classical* value  $\text{val}(G^{\otimes n})$  (i.e. where the players are restricted to using classical strategies) of a repeated game  $G^{\otimes n}$  goes down exponentially with  $n$ , for large enough  $n$  [19, 12]. This result is known as the Parallel Repetition Theorem, and is central in the study of hardness of approximation, probabilistically checkable proofs, and hardness amplification in classical theoretical computer science.

Recently, quantum analogues of the Parallel Repetition Theorem have been studied, and for certain types of games, it has been shown that the entangled game value also goes down exponentially with the number of repetitions. In particular, parallel repetition theorems have been shown for 2-player *free games* (see [6, 7, 13]) and *projection games* (see [9]). Free games are where the input distribution to the players is a product distribution (i.e. each players’ questions are chosen independently of each other). Projection games are where, for each answer of one designated player, there is at most one other answer for the other player that the referee would accept.

Most relevant to this work are the results of [6, 7, 13] on free entangled games. Among them, the best parallel repetition theorem was obtained by [7], who prove that for a two-player free game  $G$ , the entangled value of the  $n$ -fold repetition is at most  $(1 - \epsilon^2)^{\Omega(n/s)}$ , where  $s$  is the answer length of the players. When  $G$  is also a projection game, they obtain *strong parallel repetition*: the repeated game value is at most  $(1 - \epsilon)^{\Omega(n)}$ . The centerpiece of their analysis is a novel connection between communication complexity and parallel repetition of games.

## 1.1 Our results

In this work, we further develop this connection between games and communication protocols to obtain improved parallel repetition theorems for free entangled games. We present a generic framework where one obtains parallel repetition theorems for free games by designing succinct communication protocols. The core concept we present is:

*Better parallel repetition theorems from better communication protocols.*

The first instantiation of this concept is the following theorem:

► **Theorem 1.1.** *Let  $k \geq 2$  be an integer. Let  $G$  be a  $k$ -player free game with entangled value  $\text{val}^*(G) = 1 - \epsilon$ . Then, for  $n = \Omega(sk^4 \log(k/\epsilon)/\epsilon^{3/2})$ ,*

$$\text{val}^*(G^{\otimes n}) \leq (1 - \epsilon^{3/2})^{\Omega(n/k^4 s)}$$

where  $s$  is the output answer length of the players.

The proof of Theorem 1.1 uses a quantum communication protocol that performs a version of distributed unstructured search (i.e. searching for a 1 in a bitstring). The improvement of the base from  $1 - \epsilon^2$  (as found in [7]) to  $1 - \epsilon^{3/2}$  comes from the fact that the unstructured search problem on  $N$  bits can be solved by a quantum algorithm using only  $O(\sqrt{N})$  queries. We discuss this in more detail in the next section.

Our second theorem handles a broader class of games, where the players can output quantum states as answers. We are able to handle this broader class of games because our framework allows general quantum communication protocols.

► **Theorem 1.2.** *Let  $k \geq 2$  be an integer. Let  $G$  be a  $k$ -player free game, where players can output (possibly entangled) quantum states, and has entangled value  $\text{val}^*(G) = 1 - \epsilon$ . Then, for all  $n$ ,*

$$\text{val}^*(G^{\otimes n}) \leq (1 - \epsilon^2)^{\Omega(n/k^2 s)}$$

where  $s = \max_j \log(d_j)$ , where  $d_j$  is the dimension of player  $j$ 's output state.

Furthermore, we prove that the dependence of the exponent on the number of players  $k$  is necessary:

► **Theorem 1.3.** *For all  $k \geq 2$ , there exists a  $k$ -player free game  $G$  and  $n > 1$  where  $\text{val}^*(G^{\otimes n}) \geq \text{val}^*(G)^{n/k}$ .*

To our knowledge, our results are the first to show quantum parallel repetition in the setting of games with more than 2 players.

Finally, we give a proof of parallel repetition for the *classical* value of  $k$ -player free games. While this theorem appears to be a folklore result, we were not able to find any explicit proof of it. We provide one here for the sake of completeness.

► **Theorem 1.4.** *Let  $G$  be a  $k$ -player free game with classical value  $\text{val}(G) = 1 - \epsilon$ . Then*

$$\text{val}(G^{\otimes n}) \leq (1 - \epsilon^2)^{\Omega(n/sk)},$$

where  $s$  is the output answer length of the players.

**CQ Games.** Our second theorem applies to a class of games that is a generalization of the traditional notion of games that involve two players and have classical inputs and outputs. In this paper we introduce the class of  $k$ -player *classical-quantum* (CQ) games, where the players receive classical inputs, apply local unitary operators to their share of an entangled state, and return some qubits to the referee. The referee then makes a measurement on the answer qubits to decide whether to accept or reject. If we restrict the players' unitaries to be permutation matrices, and the referee's measurement to be diagonal in the standard basis, then we recover the class of classical games.

We believe the model of CQ games is worth deeper investigation. One motivation for the study of CQ games comes from the recent exciting work of Fitzsimons and Vidick [11], who demonstrated an efficient reduction transforming a local Hamiltonian  $H = H_1 + \dots + H_m$  acting on  $n$  qubits to a 5-player CQ-game  $G_H$  such that approximating  $\text{val}^*(G_H)$  with inverse

polynomial accuracy will decide whether the ground state energy of  $H$  is a YES or NO instance of the QMA-complete problem LOCAL HAMILTONIANS. In this game, the referee sends  $O(\log n)$ -sized questions, and the players respond with  $O(1)$ -qubit states as answers. The significance of this is that it opens up the possibility of proving a “games” version of the Quantum PCP conjecture. This intriguing possibility calls for further study of the behavior of CQ games.

## 1.2 Parallel repetition and communication protocols

At a high level, most proofs of parallel repetition proceed via reduction. Let  $G$  be a two-player free game with verification predicate  $V(x, y, a, b)$ . If there were a strategy  $\mathcal{S}$  for the repeated game  $G^{\otimes n}$  that wins with too large probability, then one can transform  $\mathcal{S}$  to a strategy  $\mathcal{T}$  to play a single instance of the game  $G$  with probability larger than  $\text{val}^*(G)$ , which would be a contradiction.

In [6, 7, 13], the reduction from a repeated game strategy to a single game strategy has two steps: (1) a “too-good” repeated game strategy  $\mathcal{S}$  is converted to an *advice-based strategy* for game  $G$ , which wins with high probability. An advice-based strategy is a collection of *advice states*  $\{\varphi_{xy}\}_{xy}$  so that when Alice and Bob receive inputs  $x$  and  $y$ , they happen to share the entangled state  $\varphi_{xy}$ , which they can measure to produce answers. Of course, this is not a valid quantum strategy for game  $G$ , but (2) using the assumption that  $\mathcal{S}$  has very high winning probability, the advice-based strategy can be *rounded* to a true game strategy: Alice and Bob can apply local operations  $U_x$  and  $V_y$ , respectively, on some input-independent state  $\varphi$  to approximate  $\varphi_{xy}$ , and thus simulate the advice-based strategy (with some error).

One can construct the advice states  $\{\varphi_{xy}\}_{xy}$  from  $\mathcal{S}$  in different ways. Generally, the goal is to create advice states that closely mimic the joint state of the players during an actual execution of the strategy  $\mathcal{S}$ , *conditioned* on some event. Ideally, we would like to condition on the event that the players won all  $n$  coordinates of  $G^{\otimes n}$ . This would give rise to the ideal advice states: whenever the players receive an input  $x$  and  $y$ , their advice state  $\varphi_{xy}$  would have precisely the correct answers  $a$  and  $b$  that would allow them to win the single game  $G$ .

However, it seems impossible to argue that such an ideal advice-based strategy can be simulated by a true game strategy. The approach taken by [6, 7] is to construct advice states  $\{\varphi_{xy}\}$  that have two properties: (a) the advice-based strategy succeeds with high probability for  $G$ , and (b) there is a *low-cost communication protocol* between Alice and Bob that produces  $\varphi_{xy}$  when they receive inputs  $x$  and  $y$ , respectively. Property (b) makes it possible to approximate the advice-based strategy using a valid quantum strategy: small communication complexity translates into small rounding error.

The communication protocol used in [6, 7] is a simple one: Alice and Bob first play the optimal strategy  $\mathcal{S}$  for  $G^{\otimes n}$ . They receive inputs  $(x_1, \dots, x_n)$  and  $(y_1, \dots, y_n)$ , and measure a shared state  $|\xi\rangle$  and obtain  $n$ -tuples of outputs  $(a_1, \dots, a_n)$  and  $(b_1, \dots, b_n)$ . Then, Alice samples a small subset of coordinates  $i_1, \dots, i_h$ , and sends over her inputs and outputs in this subset to Bob, who verifies that the original game  $G$  was won in each of these coordinates. If Bob finds a  $i_j$  such that the tuple  $V(x_{i_j}, y_{i_j}, a_{i_j}, b_{i_j}) = 0$ , then Bob aborts. Otherwise, Bob accepts. If we condition the final state of the protocol on Bob accepting, then we have a very good proxy for the ideal advice states described above. But since the communication complexity of this protocol is small, we can round this to a valid quantum strategy.

However, the analysis of [6, 7] is tailored to simple one-way communication protocols involving classical messages. We generalize this paradigm to show that, if the advice states  $\{\varphi_{xy}\}$  can be constructed using *any* communication protocol (which can be two-way, and involve quantum messages), then the advice-based strategy using  $\{\varphi_{xy}\}$  can be simulated

with a true game strategy, with error that is related to the communication complexity of the protocol. This unlocks a richer toolbox for the reduction designer: one can use many more tools from communication complexity to engineer good advice states. This gives rise to the concept of “Better parallel repetition theorems from better communication protocols.”

Our theorems are instantiations of this mantra. At the heart of the communication protocol used in our first theorem is a variant of the Grover search algorithm. There, the players sample a random subset of coordinates  $i_1, \dots, i_h$  as before, but now they perform quantum search over the indices to find a “losing coordinate”: i.e., a coordinate  $i_j$  such that  $V(x_{i_j}, y_{i_j}, a_{i_j}, b_{i_j}) = 0$ . The quadratic speedup of Grover’s search algorithm translates into a quadratic savings in communication complexity, which is precisely what allows us to improve the base of the repeated game value from  $1 - \epsilon^2$  to  $1 - \epsilon^{3/2}$ . For our second theorem, we take advantage of the fact that the communication protocol can be quantum, which allows us to handle games with quantum outputs.

Our use of quantum search in the protocol to generate the advice states gives a generic way to improve the reduction for arbitrary free games. However, one could also use this technique to prove *game-specific* parallel repetition theorems. That is, one could try to leverage special properties of a particular game to design a succinct communication protocol for generating advice states, and in turn, obtain a parallel repetition theorem with better parameters. Indeed, one can see this idea in the result of [7] for free projection games: by using the projection property of the game, their communication protocol avoids sending whole input and output symbols. This allows them to prove a repeated game value of  $(1 - \epsilon)^{\Omega(n)}$  – note that this does not depend on the output alphabet!

### 1.3 Related work

We discuss how our result relates to prior results in parallel repetition, classical and quantum. Most relevant to our work are the results on free games. Jain, et al. [13] and Chailloux and Scarpa [6, 7] both proved that the entangled value of 2-player free games (with classical inputs and outputs) goes down exponentially with the number of repetitions. In particular, [7] showed for such a game  $G$  with  $\text{val}^*(G) = 1 - \epsilon$ , we have that  $\text{val}^*(G^{\otimes n}) \leq (1 - \epsilon^2)^{\Omega(n/s)}$ , where  $s$  is the output length of the players. They also show that, when  $G$  is also a projection game, *strong* parallel repetition holds:  $\text{val}^*(G^{\otimes n}) \leq (1 - \epsilon)^{\Omega(n)}$ .

In a different line of work, Dinur, Steurer and Vidick show that projection games (with an arbitrary input distribution) also have an exponential decay in entangled value under parallel repetition: if  $G$  be a 2-player projection game with classical inputs and outputs, and  $\text{val}^*(G) = 1 - \epsilon$ , then  $\text{val}^*(G^{\otimes n}) \leq (1 - \epsilon^{12})^{\Omega(n)}$  [9]. This result is not comparable with our work, nor with the work of [7, 13]. While [9] can handle games with arbitrary input distributions, the games need to satisfy the projection property. On the other hand, the results on free games can handle arbitrary verification predicates, but the input distributions need to be product.

There is a rich history of study of parallel repetition in classical theoretical computer science, which we will not detail here. Most relevant to us is the work of Barak, et al. [3], who showed that for 2-player free games  $G$  with classical value  $1 - \epsilon$ ,  $\text{val}(G^{\otimes n}) \leq (1 - \epsilon^2)^{\Omega(n/s)}$ , where  $s$  is the output length of the players. Intriguingly, it is not known whether the  $\epsilon^2$  term is tight for free games (it is known that this is necessary for classical parallel repetition of general games [20]). Our first theorem demonstrates a possible separation between classical and quantum parallel repetition; the base of our repeated game value is  $1 - \epsilon^{3/2}$ , rather than  $1 - \epsilon^2$ .

Finally, there has been little prior study of the parallel repetition of games with more

than 2 players. Buhrman et al. studied this question for non-signaling players, and showed that the non-signaling value of repeated games goes down exponentially with the number of repetitions [5]. Their parallel repetition theorem holds for games with *full support*, meaning that every possible combination of questions gets asked with positive probability; furthermore, the rate of decay also depends on the complete description of the game, not just the original game value and the number of repetitions. Arnon-Friedman et al. prove similar results for multi-player non-signaling games, but they use a new technique called de Finetti reductions [2]. Rosen also studied  $k$ -player parallel repetition in a weaker version of the non-signaling model, and demonstrated an exponential rate of decay [21].

## 2 Proof overviews

### 2.1 Overview of Theorem 1.1

Here we give a very informal outline of the proof of Theorem 1.1, for the case of two-player free games. The full proof that handles an arbitrary number of players can be found in Section 5 of the Appendix.

Let  $G$  be a two-player free game, with inputs  $(x, y)$  drawn from a product distribution  $\mu = \mu_X \otimes \mu_Y$ , and with verification predicate  $V(x, y, a, b)$ . Let  $\mathcal{S}$  be an optimal strategy for the repeated game  $G^{\otimes n}$ , where Alice and Bob share an entangled state  $|\xi\rangle$ , and upon receiving a tuple of inputs  $\mathbf{x} = (x_1, \dots, x_n)$  and  $\mathbf{y} = (y_1, \dots, y_n)$ , Alice and Bob perform local measurements  $M^{\mathbf{x}} = \{M_{\mathbf{a}}^{\mathbf{x}}\}_{\mathbf{a}}$  and  $N^{\mathbf{y}} = \{N_{\mathbf{b}}^{\mathbf{y}}\}_{\mathbf{b}}$  on their respective parts of  $|\xi\rangle$ , to obtain answer tuples  $\mathbf{a} = (a_1, \dots, a_n)$  and  $\mathbf{b} = (b_1, \dots, b_n)$ . Let  $|\xi_{\mathbf{xyab}}\rangle$  be the (unnormalized) post-measurement state of  $|\xi\rangle$  after making measurements  $(M^{\mathbf{x}}, N^{\mathbf{y}})$ , and obtaining outcomes  $(\mathbf{a}, \mathbf{b})$ . We assume for contradiction that  $\text{val}^*(G^{\otimes n}) > 2^{-\gamma n}$ , for some small  $\gamma$ .

**A naive approach.** Consider the following state

$$|\theta\rangle = \frac{1}{\sqrt{\lambda}} \sum_{\mathbf{x}, \mathbf{y}} \sqrt{\mu^{\otimes n}(\mathbf{x}, \mathbf{y})} |\mathbf{x}\rangle \otimes |\mathbf{y}\rangle \otimes \sum_{\mathbf{a}, \mathbf{b}: V(\mathbf{x}, \mathbf{y}, \mathbf{a}, \mathbf{b})=1} |\xi_{\mathbf{xyab}}\rangle \otimes |\mathbf{a}\rangle \otimes |\mathbf{b}\rangle$$

where  $\lambda$  is a normalizing constant,  $\mu^{\otimes n}$  is the input distribution for  $G^{\otimes n}$ , and  $V(\mathbf{x}, \mathbf{y}, \mathbf{a}, \mathbf{b}) = \prod_i V(x_i, y_i, a_i, b_i)$ . This would be the joint state of Alice and Bob if they received inputs  $\mathbf{x}$  and  $\mathbf{y}$  in coherent superposition, played strategy  $\mathcal{S}$ , and won the game. Here is a naive idea to use  $|\theta\rangle$  in a strategy  $\mathcal{T}$  for  $G$ : Alice and Bob share  $|\theta\rangle$ , with Alice possessing the  $|\mathbf{x}\rangle$  input register, the  $|\mathbf{a}\rangle$  output register, and half of  $|\xi_{\mathbf{xyab}}\rangle$ ; Bob possesses the  $|\mathbf{y}\rangle$  input register, the  $|\mathbf{b}\rangle$  output register, and the other half of  $|\xi_{\mathbf{xyab}}\rangle$ . When they receive inputs  $(x, y) \leftarrow \mu$ , they each measure some fixed coordinate  $i$  of their respective input registers of  $|\theta\rangle$  to obtain input symbols  $x'$  and  $y'$ , respectively. Suppose that  $x = x'$  and  $y = y'$ : their shared state has collapsed to  $|\theta_{xy}\rangle$ . Then measuring the  $i$ th coordinate of their output registers will yield outputs  $(a, b)$  such that  $V(x, y, a, b) = 1$ . Thus  $\{\theta_{xy}\}$  is an excellent set of advice states – call this ensemble the *ideal advice*. However, in general, this cannot be rounded to a valid game strategy.

**Advice states from Grover search.** Instead, we will construct another ensemble that mimicks the ideal advice, and if  $\text{val}^*(G^{\otimes n})$  is too large, can be rounded to a valid game strategy with small error. We construct a state  $|\varphi\rangle$  in steps. First, Alice and Bob start with

$$|\psi^0\rangle = \sum_{\mathbf{x}, \mathbf{y}} \sqrt{\mu^{\otimes n}(\mathbf{x}, \mathbf{y})} |\mathbf{x}\rangle \otimes |\mathbf{y}\rangle \otimes \sum_{\mathbf{a}, \mathbf{b}} |\xi_{\mathbf{xyab}}\rangle \otimes |\mathbf{a}\rangle \otimes |\mathbf{b}\rangle.$$

Note that this state can be produced without any communication. Then, Alice and Bob engage in a short communication protocol to determine whether they've lost or won the game repeated: they need to determine whether there exists a coordinate  $i$  such that  $V(\mathbf{x}_i, \mathbf{y}_i, \mathbf{a}_i, \mathbf{b}_i) = 0$  – call this a *losing coordinate*. Classically, this would require  $\Omega(n)$  bits of communication, which is too large for us. Instead, Alice and Bob can perform a distributed version of Grover's algorithm to search for a losing coordinate. Although Grover's search algorithm is a quantum query algorithm, it is a standard technique to convert query algorithms into communication protocols (see [4]): Alice executes the Grover search algorithm, and whenever she has to query the  $i$ th coordinate, Alice sends the query request to Bob, who responds with  $(\mathbf{y}_i, \mathbf{b}_i)$ . Alice can then compute  $V(\mathbf{x}_i, \mathbf{y}_i, \mathbf{a}_i, \mathbf{b}_i)$ . If Alice finds a losing coordinate, she aborts the protocol. Otherwise, she accepts. Since the Grover algorithm requires  $O(\sqrt{n})$  queries, this communication protocol uses  $\tilde{O}(\sqrt{n})$  qubits of communication, where  $\tilde{O}(\cdot)$  hides the  $\log n$  bits needed for the query request, as well as the input and output lengths. This protocol is performed coherently with the  $|\mathbf{x}\rangle$ ,  $|\mathbf{y}\rangle$ ,  $|\mathbf{a}\rangle$ , and  $|\mathbf{b}\rangle$  registers. Let  $|\psi\rangle$  denote the final state of this protocol, and let  $|\varphi\rangle$  denote  $|\psi\rangle$  conditioned on Alice accepting.

If Grover search worked perfectly, then  $|\varphi\rangle$  would be essentially the same as the naive  $|\theta\rangle$  we described first. However, Grover's algorithm does not perform search perfectly, and has some error. Furthermore, when we condition on Alice not finding a losing coordinate, this error gets multiplied by  $1/\text{val}^*(G^{\otimes n})$ . Though we are assuming  $\text{val}^*(G^{\otimes n})$  is “large”, it is still exponentially small, and hence we require that the Grover search has exponentially small error. In our proof, we make some technical adjustments to the search protocol in order to handle this exponential blowup of the Grover error (without increasing the communication complexity to  $\Omega(n)$  bits), but for the sake of exposition we will ignore this issue. For now, we can treat  $|\varphi\rangle$  as a very good approximation of  $|\theta\rangle$  – thus, defining  $|\varphi_{xy}\rangle$  in the same way we defined  $|\theta_{xy}\rangle$  yields a good ensemble of advice states  $\{\varphi_{xy}\}$ .

**Rounding to a valid quantum strategy.** Now it remains to show that  $\{\varphi_{xy}\}$  can be rounded to a valid quantum strategy. We do this by establishing two properties of the state  $|\varphi\rangle$ : there exists a coordinate  $i \in [n]$  such that

1. Suppose we decohere (i.e. measure) the  $i$ th coordinate of the  $|\mathbf{xy}\rangle$  registers of  $|\varphi\rangle$ , and let  $(X_i, Y_i)$  denote the random measurement outcomes. Then the distribution of  $(X_i, Y_i)$  is  $\gamma$ -close to  $\mu$ , the input distribution of  $G$ ; and
2. Let  $B$  denote the part of  $\varphi$  controlled by Bob. Then the quantum mutual information between  $X_i$  (after measurement) and  $B$  in  $\varphi$ , denoted by  $I(X_i : B)_\varphi$ , is at most  $\gamma$ . Similarly, we have  $I(Y_i : A)_\varphi \leq \gamma$ , where  $Y_i$  and  $A$  are defined analogously.

Property 1 follows from the fact that the distribution of  $\mathbf{x}$  and  $\mathbf{y}$ , before conditioning, is a product distribution across coordinates  $i$ . When we condition on an event with probability  $\lambda$ , then on average, the distribution of the individual coordinates  $(x_i, y_i)$  are skewed by at most  $\sqrt{\log(1/\lambda)/n}$  in total variation distance. This simple but useful fact is known as *Raz's Lemma* in the parallel repetition literature. Thus, if  $\lambda \gg 2^{-n}$ , then the input distribution of most coordinates, even after conditioning, is largely unaffected. Here,  $\lambda$  corresponds to the probability that Alice does not abort the protocol, which is at least  $\text{val}^*(G^{\otimes n})$ .

Property 2 is the most interesting part of our proof. It states that Bob's part of the state  $\varphi$  is relatively uncorrelated with the value of the input  $X_i$ , and similarly Alice's part of  $\varphi$  is relatively uncorrelated with the value of  $Y_i$ . This uses the fact that our protocol has low communication complexity: intuitively, since Alice and Bob communicate at most  $\tilde{O}(\sqrt{n})$  qubits in the protocol, they “learn” at most  $\tilde{O}(\sqrt{n})$  bits total about each other's inputs.

Amortized over the  $n$  coordinates, this means Alice has about  $1/\sqrt{n}$  bits of information about each  $y_i$ , on average, and similarly for Bob. When we condition on Alice not aborting, each player's knowledge of the other's inputs increases by at most  $\log 1/\lambda$ . If  $\lambda > 2^{-\gamma n}$ , Alice's state has  $O(\gamma)$  mutual information with each  $y_i$ , on average.

This intuition is formalized by leveraging the beautiful result of Nayak and Salzman that gives limits on the ability of entanglement-assisted quantum communication protocols to transmit classical messages [17]. More specifically, consider a general two-way quantum communication protocol between Alice and Bob, who may start with some shared entangled state. Suppose that Alice is given a uniformly random  $m$ -bit message  $X$  at the beginning of the protocol. If  $T$  qubits are exchanged between Alice and Bob over the course of the protocol, Bob can only guess Alice's input  $X$  with probability at most  $2^{2T}/2^m$ . Equivalently, the mutual information between Bob's final state and  $X$  is at most  $2T$ . Applying the Nayak-Salzman theorem to our setting, and using what we call *Quantum Raz's Lemma*<sup>1</sup>, we can conclude that on average,  $I(Y_i : A)_\varphi = I(X_i : B)_\varphi = \frac{1}{n} (\tilde{O}(\sqrt{n}) + \log 1/\lambda) = O(\gamma)$ .

Once we have established Property 1 and 2, then the *Quantum Strategy Rounding Lemma* (which can be found in both [6, 13]) then gives that there exists a unitaries  $\{U_x\}_x$  and  $\{V_y\}_y$  for Alice and Bob, respectively, so that  $U_x \otimes V_y |\varphi\rangle \approx |\varphi_{xy}\rangle$ . Thus we have a valid quantum strategy for  $G$ : on input  $(x, y)$ , Alice and Bob locally apply unitaries  $U_x$  and  $V_y$  to their shared state  $\varphi$ , and obtain something close to the advice state  $\varphi_{xy}$ , which they can use to win game  $G$  with probability close to 1. For sufficiently small  $\gamma$ , this will be greater than  $\text{val}^*(G)$ , a contradiction. Thus,  $\text{val}^*(G^{\otimes n}) \leq 2^{-\gamma n}$ . This concludes the proof outline.

**Other technical considerations.** While this discussion has been very informal, it captures the conceptual arguments that are required by our analysis. There are many technical details that are handled by the full proof in the Appendix: for example, in order to make the error in the Grover search exponentially small, we increase the communication complexity of the protocol to  $\tilde{O}(\log(1/\lambda)/\sqrt{\epsilon})$ . If  $\lambda < 2^{-\epsilon^{3/2}n}$ , where  $\text{val}^*(G) = 1 - \epsilon$ , then the communication complexity is at most  $\tilde{O}(\epsilon n)$ , which is still small enough for use in Quantum Raz's Lemma. Another issue is that the communication protocol described requires that Bob transmit his input symbols  $y_i$ , which would incur a dependence on the input alphabet size. Through a modification of the protocol and the analysis, we are able to avoid this dependence. Finally, instead of using Grover's algorithm exactly, we use the 3-dimensional search algorithm of Aaronson and Ambainis [1], which performs quantum search in a "spatially local" way. When converted to a communication protocol, the parties no longer need to incur a  $\log n$ -qubit overhead per round simply to transmit a query request.

The arguments above are not specific to two-players. We prove our theorem for the general  $k$ -player case. An important part of this is the  $k$ -player generalization of the Quantum Strategy Rounding lemma of [6, 13], which we prove in Lemma 4.3.

## 2.2 Overview of Theorem 1.2

Theorem 1.2 shows parallel repetition for the entangled value of  $k$ -player free CQ games: these are games where the players may produce quantum states as answers. Instead of

<sup>1</sup> We make a quick remark about Quantum Raz's Lemma. The ingredients of Quantum Raz's Lemma can be found in various forms in [6, 7, 13], but we find it conceptually advantageous to consolidate these ingredients into a single Lemma that is used as a black box. The benefit of this consolidation is that the overarching structure of the proofs of Theorems 1.1 and 1.2 are the same – really, the only essential difference is the communication protocol!



making measurements on their share of the entangled state, players will apply local unitaries (that depend on their inputs), and transmit a number of qubits to the referee. The referee will then perform some joint verification measurement on all the answer qubits to decide whether to accept or not.

Similarly to the proof of Theorem 1.1, we will use a low-cost communication protocol to design advice states for the repeated-game-to-single-game reduction. However, we were not able to use the distributed Grover search technique. Instead, our low-cost communication protocol performs the following (in the two-player setting): Alice will send Bob her inputs and answer qubits corresponding to coordinates in a small subset  $C \subseteq [n]$ . Bob will then perform the referee's verification measurement on Alice's inputs and outputs, and his own inputs and outputs, to determine whether they won all the coordinates in the subset  $C$ . Having determined whether they won or not, Bob will return Alice's message back to her. If  $C$  is sufficiently small, then the communication cost of this task is small.

This simple checking protocol is similar to the checking protocol of [7]. However, in our protocol, Alice and Bob exchange quantum messages, and it is a two-way protocol (because Bob has to return Alice's message back to her). The theorem of [17] again allows us to show that  $I(X_i : B)_\varphi$  and  $I(Y_i : A)_\varphi$  are small, where  $\varphi$  is the advice state that arises from this communication protocol, and thus we can apply quantum strategy rounding as before.

For the proofs of Theorems 1.2, 1.3, and 1.4, we refer the reader to the full version of our paper [8].

**Outline.** In Section 3, we list the quantum information theoretic facts we'll need, as well as prove a few useful technical lemmas (including Quantum Raz's Lemma). In Section 4, we prove our  $k$ -player Quantum Strategy Rounding lemma. We prove Theorem 1.1 in Section 5.

### 3 Preliminaries

We assume familiarity with the basics of quantum information and computation. For a comprehensive reference, we refer the reader to [18, 22]. For a pure state  $|\psi\rangle$ , we will let  $\psi$  denote the density matrix  $|\psi\rangle\langle\psi|$ . If  $|\psi\rangle^{AB}$  is a bipartite state, then  $\psi^A$  will be the reduced density matrix of  $\psi^{AB}$  on space  $A$ . A density matrix  $\rho^{XA}$  is a *classical-quantum* (CQ) state if  $\rho^{XA} = \sum_x p(x) |x\rangle\langle x| \otimes \rho_x^A$ , where  $p(x)$  is a probability distribution and  $\rho_x^A$  is an arbitrary density matrix on space  $A$ . For a probability distribution  $\mu$ ,  $x \leftarrow \mu$  indicates  $x$  is drawn from  $\mu$ . For a classical state  $\rho^X = \sum_x \mu(x) |x\rangle\langle x|$ , we write  $x \leftarrow \rho^X$  to denote  $x \leftarrow \mu$ . We let  $\text{id}$  denote the identity matrix.

#### 3.1 $k$ -player games

We give a formal definition of games, where the inputs and outputs are classical (in the full version of this paper, we give a more general definition of *CQ games*, where the outputs may be quantum [8]). A  $k$ -player game is a tuple  $G = (\mathcal{X}, \mathcal{A}, \mu, V)$ , where:

1.  $\mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_k$  with each  $\mathcal{X}_j$  a finite alphabet,
2.  $\mathcal{A} = \mathcal{A}_1 \times \cdots \times \mathcal{A}_k$  with each  $\mathcal{A}_j$  a finite alphabet,
3.  $\mu$  is a distribution over  $\mathcal{X}$ ,
4.  $V : \mathcal{X} \times \mathcal{A} \rightarrow \{0, 1\}$  is the verification predicate.

In a  $k$ -player  $G$ , a referee samples an input  $x = (x_1, \dots, x_k)$  from  $\mu$ , and sends  $x_j$  to player  $j$ . The players produce a vector of outputs  $a = (a_1, \dots, a_k)$  (where the  $j$ th player outputs symbol  $a_j$ , and the referee accepts if  $V(x, a) = 1$ ).

We say a game is *free* if  $\mu = \mu_1 \otimes \cdots \otimes \mu_k$ , where  $\mu_j$  is a distribution over  $\mathcal{X}_j$  (i.e.  $\mu$  is a product distribution). A *quantum strategy* for  $G$  is a shared state  $|\xi\rangle^E$  (where  $E$  are  $k$ -partite spaces split between the  $k$  players), and for each player  $j$  a set of measurements  $\{M^{j,x_j}\}_{x_j \in \mathcal{X}_j}$  (with each  $M^{j,x_j}$  being a set of POVM elements  $\{M_{a_j}^{j,x_j}\}_{a_j \in \mathcal{A}_j}$ ) which act on the space  $E_j$ . On input  $x_j$ , player  $j$  measures the  $E_j$  register of  $|\xi\rangle$  using measurement  $M^{j,x_j}$ , and obtains an outcome  $a_j$ , which is then sent to the referee. The entangled value of a game  $G$  is defined as the maximum probability a referee will accept over all possible (finite-dimensional) quantum strategies for  $k$  players:

$$\text{val}^*(G) = \max_{|\xi\rangle, \{M^{j,x_j}\}_{x_j \in \mathcal{X}_j}} \mathbb{E}_{x \leftarrow \mu} \left[ \sum_{a \in \mathcal{A}: V(x,a)=1} \text{tr} (M_{a_1}^{1,x_1} \otimes \cdots \otimes M_{a_k}^{k,x_k} \xi) \right].$$

The  $n$ -fold repetition of a game  $G = (\mathcal{X}, \mathcal{A}, \mu, V)$  is denoted by  $G^{\otimes n} = (\mathcal{X}^n, \mathcal{A}^n, \mu^{\otimes n}, V^n)$ , where:  $\mu^{\otimes n}$  is the product distribution over  $n$  independent copies of  $\mathcal{X}$ , and  $V^n(\vec{x}, \vec{a}) := \prod_i V(x_i, a_i)$ , with  $\vec{x} \in \mathcal{X}^n$  and  $\vec{a} \in \mathcal{A}^n$ .

### 3.2 Properties of the squared Bures metric

For two positive semidefinite operators  $\rho, \sigma$ , let the fidelity between  $\rho$  and  $\sigma$  be denoted by  $F(\rho, \sigma) := \text{tr} \sqrt{\rho^{1/2} \sigma \rho^{1/2}}$ . The fidelity distance measure has the well-known property that for pure states  $|\psi\rangle$  and  $|\varphi\rangle$ ,  $F(\psi, \varphi) = |\langle \psi | \varphi \rangle|$ . Furthermore, when  $\rho$  and  $\sigma$  are classical probability distributions in the same basis (i.e.  $\rho = \sum_i p_i |i\rangle\langle i|$  and  $\sigma = \sum_i q_i |i\rangle\langle i|$ ), then  $F(\rho, \sigma) = \sum_i \sqrt{p_i q_i}$ .

The fidelity distance measure is not a metric on the space of positive semidefinite operators. For one, it does not satisfy a triangle inequality. However, one can convert fidelity into other measures that are metrics. One such measure is the *Bures metric*, defined as  $\mathcal{B}(\rho, \sigma) := \sqrt{1 - F(\rho, \sigma)}$ . In this paper, we will use the *squared Bures metric*, denoted by  $K(\rho, \sigma) := \mathcal{B}(\rho, \sigma)^2$ , as the primary distance measure between quantum states. It satisfies many pleasant properties, including the following:

► **Fact 3.1** (Triangle inequality). *Let  $n \geq 2$  and let  $\rho_1, \dots, \rho_{n+1}$  be density matrices. Then*

$$K(\rho_1, \rho_{n+1}) \leq n \sum_i K(\rho_i, \rho_{i+1}).$$

**Proof.** We adapt the proof from [7]. For  $i \in [n]$  let  $\alpha_i = \arccos(F(\rho_i, \rho_{i+1}))$ . Let  $\alpha = \arccos(F(\rho_1, \rho_{n+1}))$ . Then, since  $\arccos(F(\cdot, \cdot))$  is a distance measure for quantum states, we have  $\alpha \leq \sum_i \alpha_i$ . Then we have

$$K(\rho_1, \rho_{n+1}) = 1 - \cos(\alpha) \leq n^2(1 - \cos(\alpha/n)) \leq n \sum_i (1 - \cos(\alpha_i)) = n \sum_{i=1}^n K(\rho_i, \rho_{i+1}).$$

◀

► **Fact 3.2** (Contractivity under quantum operations). *Let  $\mathcal{E}$  be a quantum operation, and let  $\rho$  and  $\sigma$  be density matrices. Then  $K(\mathcal{E}(\rho), \mathcal{E}(\sigma)) \leq K(\rho, \sigma)$ .*

► **Fact 3.3** (Unitary invariance). *Let  $U$  be unitary, and let  $\rho$  and  $\sigma$  be density matrices. Then  $K(U\rho U^\dagger, U\sigma U^\dagger) = K(\rho, \sigma)$ .*

► **Fact 3.4** (Convexity). *Let  $\{A_i\}$  and  $\{B_i\}$  be finite collections of positive semidefinite operators, and let  $\{p_i\}$  be a probability distribution. Then  $K(\sum_i p_i A_i, \sum_i p_i B_i) \leq \sum_i p_i K(A_i, B_i)$ .*

► **Fact 3.5.** Let  $\{A_i\}$  and  $\{B_i\}$  be finite collections of positive semidefinite operators, and let  $\{p_i\}$  be a probability distribution. Then  $K(\sum_i p_i |i\rangle\langle i| \otimes A_i, \sum_i p_i |i\rangle\langle i| \otimes B_i) = \sum_i p_i K(A_i, B_i)$ .

### 3.3 Quantum information theory

For two positive semidefinite operators  $\rho, \sigma$ , the relative entropy  $S(\rho\|\sigma)$  is defined to be  $\text{tr}(\rho(\log \rho - \log \sigma))$ . The relative min-entropy  $S_\infty(\rho\|\sigma)$  is defined as  $\min\{\lambda : \rho \preceq 2^\lambda \sigma\}$ . The entropy of  $\rho$  is denoted by  $H(\rho) := -\text{tr}(\rho \log \rho)$ . For a tripartite state  $\rho^{ABC}$ , the conditional mutual information  $H(A|B)_\rho$  is defined as  $H(\rho^{AB}) - H(\rho^B)$ . Let  $\rho^{AB}$  be a bipartite state. Then the mutual information  $I(A : B)_\rho$  is defined as  $H(A)_\rho - H(A|B)_\rho$ . An equivalent definition is  $I(A : B)_\rho = S(\rho^{AB}\|\rho^A \otimes \rho^B)$ .

► **Fact 3.6** ([14]). Let  $\rho$  and  $\sigma$  be density matrices. Then  $S(\rho\|\sigma) \geq K(\rho, \sigma)$ .

► **Fact 3.7** ([14]). Let  $\mu$  be a probability distribution on  $\mathcal{X}$ . Let  $\rho = \sum_{x \in \mathcal{X}} \mu_x |x\rangle\langle x| \otimes \rho_x^A$ . Then  $I(X : A)_\rho = \mathbb{E}_{x \leftarrow \mu} [S(\rho_x\|\rho)]$ .

► **Fact 3.8** ([13], Fact II.11). Let  $\rho^{XY}$  and  $\sigma^{XY}$  be quantum states. Then  $S(\rho^{XY}\|\sigma^{XY}) \geq S(\rho^X\|\sigma^X)$ .

► **Fact 3.9.** Let  $\rho^{XY}$  and  $\sigma^{XY} = \sigma^X \otimes \sigma^Y$  be quantum states. Then  $S(\rho^{XY}\|\sigma^{XY}) \geq S(\rho^X\|\sigma^X) + S(\rho^Y\|\sigma^Y)$ .

► **Fact 3.10** ([13], Fact II.8). Let  $\rho = \sum_x \mu(x) |x\rangle\langle x| \otimes \rho_x$ , and  $\rho^1 = \sum_x \mu^1(x) |x\rangle\langle x| \otimes \rho_x^1$ . Then  $S(\rho^1\|\rho) = S(\mu^1\|\mu) + \mathbb{E}_{x \leftarrow \mu^1} [S(\rho_x^1\|\rho_x)]$ .

► **Fact 3.11** ([13], Lemma II.13). Let  $\rho = p\rho_0 + (1-p)\rho_1$ . Then  $S_\infty(\rho_0\|\rho) \leq \log 1/p$ .

► **Fact 3.12.** Let  $\rho^{AB}$  and  $\sigma^{AB}$  be density matrices. Then  $S_\infty(\rho^{AB}\|\sigma^{AB}) \geq S_\infty(\rho^A\|\sigma^B)$ .

► **Fact 3.13.** Let  $\rho, \sigma$ , and  $\tau$  be density matrices such that  $S_\infty(\rho\|\sigma) \leq \lambda_1$  and  $S_\infty(\sigma\|\tau) \leq \lambda_2$ . Then  $S_\infty(\rho\|\tau) \leq \lambda_1 + \lambda_2$ .

► **Fact 3.14.** Let  $\rho, \sigma$ , and  $\tau$  be density matrices such that  $S(\rho\|\sigma) \leq \lambda_1$  and  $S_\infty(\sigma\|\tau) \leq \lambda_2$ . Then  $S_\infty(\rho\|\tau) \leq \lambda_1 + \lambda_2$ .

**Proof.**  $S_\infty(\sigma\|\tau) = \lambda_2$  implies that  $2^{-\lambda_2} \sigma \preceq \tau$ . Then,

$$\begin{aligned} S(\rho\|\tau) &= \text{tr}(\rho(\log \rho - \log \tau)) \\ &\leq \text{tr}(\rho(\log \rho - \log 2^{-\lambda_2} \sigma)) \\ &\leq \text{tr}(\rho(\log \rho - (-\lambda_2)\text{id} - \log \sigma)) \\ &\leq \lambda_2 + \text{tr}(\rho(\log \rho - \log \sigma)) \\ &= \lambda_1 + \lambda_2. \end{aligned}$$

◀

### 3.4 Some technical lemmas

The following lemma is due to [3]:

► **Lemma 3.15** ([3], Lemma 3.3). Let  $P = (p, 1-p)$  and  $Q = (q, 1-q)$  be binary distributions. If  $S(P\|Q) \leq \delta$ , and  $p < \delta$ , then  $q \leq 4\delta$ .

The following adapts Lemma 3.15 to use the distance measure  $K$  instead:

► **Lemma 3.16.** *Let  $P = (p, 1-p)$  and  $Q = (q, 1-q)$  be binary distributions. If  $K(P, Q) \leq \delta$ , and  $p < \delta$ , then  $q \leq 9\delta$ .*

**Proof.** If  $q \leq p$ , then we are done. Assume otherwise. We have that  $\delta \geq K(P, Q) \geq (1 - F(P, Q)^2)/2$ , because  $0 \leq F(P, Q) \leq 1$ . Then,

$$\begin{aligned} F(P, Q)^2 &= (\sqrt{pq} + \sqrt{(1-p)(1-q)})^2 \\ &= pq + 1 - p - q + pq + 2\sqrt{pq(1-p)(1-q)}, \end{aligned}$$

and thus

$$\begin{aligned} 2\delta &\geq p + q - 2pq - 2\sqrt{pq(1-p)(1-q)} \\ &\geq p + q - 2pq - 2\sqrt{pq} \\ &= (\sqrt{p} - \sqrt{q})^2 - 2pq \\ &\geq (\sqrt{p} - \sqrt{q})^2 - 2\delta, \end{aligned}$$

where in the last line we used the assumption that  $p \leq \delta$ . Then  $2\sqrt{\delta} \geq |\sqrt{p} - \sqrt{q}|$ . Either  $q \leq p$ , in which case  $q \leq \delta$ , or  $q \geq p$ , in which case  $\sqrt{q} \leq 2\sqrt{\delta} + \sqrt{p} \leq 3\sqrt{\delta}$ , so  $q \leq 9\delta$ . ◀

Finally, we prove a quantum analogue of Raz's Lemma, which is the central tool behind many information-theoretic proofs of parallel repetition theorems [19, 12, 3]:

► **Lemma 3.17** (Quantum Raz's Lemma). *Let  $\psi^{XA} = \left(\sum_x \mu(x) |x\rangle\langle x|^X\right) \otimes \psi^A$  be a CQ-state, classical on  $X$  and quantum on  $A$ , where  $X$  is  $n$ -partite. Furthermore, suppose that  $\mu(x) = \prod \mu_i(x_i)$ . Let  $\varphi^{XA} = \sum_x \sigma(x) |x\rangle\langle x|^X \otimes \varphi_x^A$  be such that  $S(\varphi\|\psi) \leq t$ . Then,*

$$\sum_i I(X_i : A)_\varphi \leq 2t.$$

**Proof.** First observe the following manipulations:

$$\begin{aligned} t &\geq S(\varphi^{XA}\|\psi^{XA}) \\ &= S(\varphi^{XA}\|\psi^X \otimes \psi^A) \\ &\geq S(\varphi^{XA}\|\varphi^X \otimes \varphi^A) \\ &= I(X : A)_\varphi \\ &= H(X)_\varphi - H(X|A)_\varphi \\ &\geq H(X)_\varphi - \sum_i H(X_i|A)_\varphi. \end{aligned}$$

We focus on  $H(X)_\varphi$  now. Using that relative entropy is always non-negative:

$$\begin{aligned} -H(X)_\varphi + \sum_i H(X_i)_\varphi &\leq -H(X)_\varphi + \sum_i S(\varphi^{X_i}\|\psi^{X_i}) + H(X_i)_\varphi \\ &= -H(X)_\varphi - \sum_i \text{tr}(\varphi^{X_i} \log \psi^{X_i}) \\ &= -H(X)_\varphi - \text{tr}(\varphi^X \log \psi^X) \\ &= S(\varphi^X\|\psi^X) \\ &\leq t. \end{aligned}$$

Continuing, we have

$$t \geq -t + \sum_i H(X_i)_\varphi - H(X_i|A)_\varphi = -t + \sum_i I(X_i : A)_\varphi.$$

◀

#### 4 Quantum strategy rounding

In this section we prove our  $k$ -player Quantum Strategy Rounding lemma, generalizing the technique of [7, 13].

► **Lemma 4.1** ([13]). *Let  $\mu$  be a probability distribution on  $\mathcal{X}$ . Let*

$$|\varphi\rangle := \sum_{x \in \mathcal{X}} \sqrt{\mu(x)} |xx\rangle^{XX'} \otimes |\varphi_x\rangle^{AB}.$$

Let  $|\varphi_x\rangle := |xx\rangle^{XX'} \otimes |\varphi_x\rangle^{AB}$ . Then there exists unitary operators  $\{U_x\}_{x \in \mathcal{X}}$  acting on  $XX'A$  such that

$$\mathbb{E}_{x \leftarrow \mu} [K(\varphi_x, U_x \varphi U_x^\dagger)] \leq I(X : B)_\varphi.$$

**Proof.** We follow the proof in [13]. Denote the reduced states of Bob by  $\rho_x := \text{tr}_{XX'A}(\varphi_x)$  and  $\rho := \text{tr}_{XX'A}(\varphi)$ . By Facts 3.6 and 3.7, we get that

$$I(X : B)_\varphi = \mathbb{E}_{x \leftarrow \mu} [S(\rho_x \| \rho)] \geq \mathbb{E}_{x \leftarrow \mu} [K(\rho_x, \rho)].$$

By Uhlmann's Theorem, for each  $x \in \mathcal{X}$  there exists  $U_x$  such that  $|\langle \varphi_x | (U_x \otimes \text{id}_B) | \varphi \rangle| = F(\rho_x, \rho)$ . Furthermore, this is equal to  $F(\varphi_x, U_x \otimes \text{id}_B \varphi U_x^\dagger \otimes \text{id}_B)$ . We thus obtain the claim. ◀

► **Lemma 4.2.** *Let  $\{|\varphi_a\rangle\}_{a \in \mathcal{A}}$  be a finite collection of pure states. Let  $\mu$  and  $\tau$  be probability distributions over  $\mathcal{A}$  such that  $S(\mu \| \tau) \leq \epsilon$ . Then*

$$K\left(\mathbb{E}_{a \leftarrow \mu} [|\varphi_a\rangle\langle \varphi_a|], \mathbb{E}_{a \leftarrow \tau} [|\varphi_a\rangle\langle \varphi_a|]\right) \leq \epsilon.$$

**Proof.** Consider the states

$$|\psi^\mu\rangle = \sum_{a \in \mathcal{A}} \sqrt{\mu_a} |aa\rangle^{AA'} \otimes |\varphi_a\rangle$$

and

$$|\psi^\tau\rangle = \sum_{a \in \mathcal{A}} \sqrt{\tau_a} |aa\rangle^{AA'} \otimes |\varphi_a\rangle.$$

Let  $\rho^\mu = \text{tr}_{A'}(|\psi^\mu\rangle\langle \psi^\mu|)$  and  $\rho^\tau = \text{tr}_{A'}(|\psi^\tau\rangle\langle \psi^\tau|)$ . Then notice that  $\mathbb{E}_{a \leftarrow \mu} [|\varphi_a\rangle\langle \varphi_a|] = \text{tr}_{AA'}(\rho^\mu)$  and  $\mathbb{E}_{a \leftarrow \tau} [|\varphi_a\rangle\langle \varphi_a|] = \text{tr}_{AA'}(\rho^\tau)$ , respectively. We then have that, considering the partial trace as a quantum operation,  $K(\mathbb{E}_{a \leftarrow \mu} [|\varphi_a\rangle\langle \varphi_a|], \mathbb{E}_{a \leftarrow \tau} [|\varphi_a\rangle\langle \varphi_a|]) \leq K(\rho^\mu, \rho^\tau)$ . By Uhlmann's Theorem, this is at most  $1 - |\langle \psi^\mu | \psi^\tau \rangle| = 1 - \sum_{a \in \mathcal{A}} \sqrt{\mu_a \tau_a} = K(\mu, \tau)$ . By Fact 3.6, this is at most  $S(\mu \| \tau) \leq \epsilon$ . ◀

► **Lemma 4.3** (Quantum strategy rounding). *Let  $k \geq 1$ . Let  $\mu$  be a probability distribution over  $\mathcal{X} = \mathcal{X}_1 \times \mathcal{X}_2 \times \cdots \times \mathcal{X}_k$ , where the  $\mathcal{X}_i$  are finite alphabets. Let*

$$|\varphi\rangle := \sum_{x \in \mathcal{X}} \sqrt{\mu(x)} |xx\rangle^{XX'} \otimes |\varphi_x\rangle^{AB}$$

where  $X = X_1 \cdots X_k$ ,  $X' = X'_1 \cdots X'_k$ , and  $A = A_1 \cdots A_k$  are  $k$ -partite registers. Then for all  $i \in [k]$  there exist operators  $\{U_a^i\}_{a \in \mathcal{X}_i}$  acting on  $X_i X'_i A_i$  such that

$$\mathbb{E}_{x \leftarrow \mu} [K(\varphi_x, (U_{x_1}^1 \otimes \cdots \otimes U_{x_k}^k) \varphi (U_{x_1}^{1,\dagger} \otimes \cdots \otimes U_{x_k}^{k,\dagger}))] \leq 4k \sum_i I(X_i : X_{-i} X'_{-i} A_{-i} B)_\varphi,$$

where  $A_{-i}$ ,  $X_{-i}$ , and  $X'_{-i}$  denote the  $A$ ,  $X$ , and  $X'$  registers excluding the  $i$ th coordinate, respectively, and for all  $x \in \mathcal{X}$ ,  $|\varphi_x\rangle := |xx\rangle \otimes |\varphi_x\rangle$ .

**Proof.** For  $i \in [k]$ , let  $\nu_i = \mu_1 \otimes \cdots \otimes \mu_i \otimes \mu_{>i}$ , where  $\mu_j$  denotes the marginal distribution of  $\mu$  on coordinate  $j$ , and  $\mu_{>i}$  denotes the marginal distribution of  $\mu$  on coordinates  $i + 1, \dots, k$ . For  $x \in \mathcal{X}$ , for all  $S \subseteq [k]$ , let  $x_S$  denote the coordinates of  $x$  that are in  $S$ . Therefore,  $x_{\leq i} = x_{1\dots i}$ , and  $x_{>i} = x_{i+1\dots k}$ , etc. For all  $i \in [k]$  and  $x \in \mathcal{X}$ , define

$$|\varphi_{x_{>i}}\rangle := |x_{>i} x_{>i}\rangle^{X_{>i} X'_{>i}} \otimes \left( \sum_{x_{\leq i}} \sqrt{\mu(x_{\leq i} | x_{>i})} |x_{\leq i} x_{\leq i}\rangle^{X_{\leq i} X'_{\leq i}} \otimes |\varphi_x\rangle^{AB} \right)$$

and

$$|\varphi_{x_i}\rangle := |x_i x_i\rangle^{X_i X'_i} \otimes \left( \sum_{x_{-i}} \sqrt{\mu(x_{-i} | x_i)} |x_{-i} x_{-i}\rangle^{X_{-i} X'_{-i}} \otimes |\varphi_x\rangle^{AB} \right).$$

Note that for all  $i$ ,  $|\varphi\rangle = \sum_{x_i} \sqrt{\mu_i(x_i)} |\varphi_{x_i}\rangle$ . Then by Lemma 4.1, we get that there exists unitaries  $\{U_u^i\}_{u \in \mathcal{X}_i}$  acting on  $X_i X'_i A_i$  such that

$$\mathbb{E}_{x_i \leftarrow \mu_i} [K(\varphi_{x_i}, \mathcal{U}_{x_i}^i(\varphi))] \leq I(X_i : X_{-i} X'_{-i} A_{-i} B)_\varphi,$$

where  $\mathcal{U}_{x_i}^i$  is the CP map  $\sigma \mapsto U_{x_i}^i \sigma (U_{x_i}^i)^\dagger$ . Define  $|\tilde{\varphi}_{x_{>i}}\rangle = |x_{>i}\rangle^{X''_{>i}} \otimes |\varphi_{x_{>i}}\rangle$ ,  $|\tilde{\varphi}_{x_i}\rangle = |x_i\rangle^{X''_i} \otimes |\varphi_{x_i}\rangle$ , and  $|\tilde{\varphi}_x\rangle = |x\rangle^{X''} \otimes |\varphi_x\rangle$ . For notational convenience, let  $\epsilon_i = I(X_i : X_{-i} X'_{-i} A_{-i} B)_\varphi$ , and let  $x$ ,  $x_i$  and  $x_{>i}$  denote the pure states  $|x\rangle\langle x|$ ,  $|x_i\rangle\langle x_i|$ , and  $|x_{>i}\rangle\langle x_{>i}|$  respectively.

Define the following states:  $\rho_0 = \mathbb{E}_{x \leftarrow \mu} [x \otimes \varphi_x] = \mathbb{E}_{x \leftarrow \mu} [\tilde{\varphi}_x]$ , and for all  $i \in [k]$ ,  $\rho_i = \mathbb{E}_{x \leftarrow \nu_i} [x \otimes \mathcal{U}_{x_{\leq i}}(\varphi_{x_{>i}})]$ , where  $\mathcal{U}_{x_{\leq i}}$  denotes the CP map  $\sigma \mapsto \left( \bigotimes_{j \leq i} U_{x_j}^j \right) \sigma \left( \bigotimes_{j \leq i} U_{x_j}^j \right)^\dagger$ . Then by the triangle inequality for the squared Bures metric (Fact 3.1),

$$K(\rho_0, \rho_n) \leq k \sum_{i=0}^{k-1} K(\rho_i, \rho_{i+1}).$$

We upper bound each term  $K(\rho_i, \rho_{i+1})$ :

$$\begin{aligned} & K \left( \mathbb{E}_{x \leftarrow \nu_i} [x \otimes \mathcal{U}_{x_{\leq i}}(\varphi_{x_{>i}})], \mathbb{E}_{x \leftarrow \nu_{i+1}} [x \otimes \mathcal{U}_{x_{\leq i+1}}(\varphi_{x_{>i+1}})] \right) \leq \\ & \mathbb{E}_{x_{\leq i} \leftarrow \otimes \mu_i} K \left( \mathcal{U}_{x_{\leq i}} \left( \mathbb{E}_{x_{>i} \leftarrow \mu_{>i}} [x_{>i} \otimes \varphi_{x_{>i}}] \right), \mathcal{U}_{x_{\leq i}} \left( \mathbb{E}_{x_{>i} \leftarrow \mu_{i+1} \otimes \mu_{>i+1}} [x_{>i} \otimes \mathcal{U}_{x_{i+1}}^{i+1}(\varphi_{x_{>i+1}})] \right) \right) \\ & = K \left( \mathbb{E}_{x_{>i} \leftarrow \mu_{>i}} [x_{>i} \otimes \varphi_{x_{>i}}], \mathbb{E}_{x_{>i} \leftarrow \mu_{i+1} \otimes \mu_{>i+1}} [x_{>i} \otimes \mathcal{U}_{x_{i+1}}^{i+1}(\varphi_{x_{>i+1}})] \right) \\ & = K \left( \mathbb{E}_{x_{>i} \leftarrow \mu_{>i}} [\tilde{\varphi}_{x_{>i}}], \mathbb{E}_{x_{i+1} \leftarrow \mu_{i+1}} \left[ x_{i+1} \otimes \mathcal{U}_{x_{i+1}}^{i+1} \left( \mathbb{E}_{x_{>i+1} \leftarrow \mu_{>i+1}} [\tilde{\varphi}_{x_{>i+1}}] \right) \right] \right) \end{aligned}$$

The second and third lines follow from the convexity and unitary invariance of the squared Bures metric, respectively (Facts 3.4 and Fact 3.3). Consider the operation  $\mathcal{E}$  that measures the

registers  $X_{>i+1} = X_{i+2} \dots X_k$  in the standard basis, and copies the outcomes into new registers  $X''_{>i+1}$ . Then  $\mathcal{E}(\mathbb{E}_{x_{i+1} \leftarrow \mu_{i+1}}[\tilde{\varphi}_{x_{i+1}}]) = \mathbb{E}_{x_{>i} \leftarrow \mu_{>i}}[\tilde{\varphi}_{x_{>i}}]$  and  $\mathcal{E}(\varphi) = \mathbb{E}_{x_{>i+1} \leftarrow \mu_{>i+1}}[\tilde{\varphi}_{x_{>i+1}}]$ . Then since  $\mathcal{E}$  commutes with  $\mathcal{U}_{x_{i+1}}^{i+1}$  and doesn't act on the  $X''_{i+1}$  register, we have that the line above is equal to

$$\begin{aligned} &= K \left( \mathcal{E} \left( \mathbb{E}_{x_{i+1} \leftarrow \mu_{i+1}}[\tilde{\varphi}_{x_{i+1}}] \right), \mathcal{E} \left( \mathbb{E}_{x_{i+1} \leftarrow \mu_{i+1}}[x_{i+1} \otimes \mathcal{U}_{x_{i+1}}^{i+1}(\varphi)] \right) \right) \\ &\leq K \left( \mathbb{E}_{x_{i+1} \leftarrow \mu_{i+1}}[\tilde{\varphi}_{x_{i+1}}], \mathbb{E}_{x_{i+1} \leftarrow \mu_{i+1}}[x_{i+1} \otimes \mathcal{U}_{x_{i+1}}^{i+1}(\varphi)] \right) \\ &= \mathbb{E}_{x_{i+1} \leftarrow \mu_{i+1}} K \left( \varphi_{x_{i+1}}, \mathcal{U}_{x_{i+1}}^{i+1}(\varphi) \right) \\ &\leq \epsilon_{i+1}. \end{aligned}$$

To complete the proof, we use the triangle inequality once more:

$$\begin{aligned} \mathbb{E}_{x \leftarrow \mu} K(\varphi_x, \mathcal{U}_x(\varphi)) &= K \left( \mathbb{E}_{x \leftarrow \mu}[\tilde{\varphi}_x], \mathbb{E}_{x \leftarrow \mu}[x \otimes \mathcal{U}_x(\varphi)] \right) \\ &\leq 2K \left( \mathbb{E}_{x \leftarrow \mu}[\tilde{\varphi}_x], \mathbb{E}_{x \leftarrow \nu_k}[x \otimes \mathcal{U}_x(\varphi)] \right) + \\ &\quad 2K \left( \mathbb{E}_{x \leftarrow \nu_k}[x \otimes \mathcal{U}_x(\varphi)], \mathbb{E}_{x \leftarrow \mu}[x \otimes \mathcal{U}_x(\varphi)] \right) \\ &\leq 2k \sum_i \epsilon_i + 2k \sum_i \epsilon_i \\ &\leq 4k \sum_i \epsilon_i. \end{aligned}$$

where  $\mathcal{U}_x$  is the composition of  $\mathcal{U}_{x_i}^i$  for all  $i \in [k]$ . Here we used Lemma 4.2 in the second line, and the fact that  $S(\mu \parallel \nu_k) = I(X_1 : X_2 : \dots : X_k)_\mu$ , which is the multipartite mutual information between the coordinates of  $X$ . It is a known fact (see, e.g., [23]) that the multipartite mutual information can be written in terms of the (standard) bipartite mutual information like so:

$$I(X_1 : X_2 : \dots : X_k)_\mu \leq I(X_1 : X_2)_\mu + I(X_1 X_2 : X_3)_\mu + \dots + I(X_1 X_2 \dots X_{k-1} : X_k)_\mu,$$

but by the data processing inequality, we have that for all  $i$ ,  $I(X_1 \dots X_{i-1} : X_i)_\mu \leq I(X_{-i} : X_i)_\mu \leq I(X_i : X_{-i} X'_{-i} A_{-i} B)_\varphi = \epsilon_i$ .  $\blacktriangleleft$

## 5 Parallel repetition using fast quantum search

**Notation.** Let  $G = (\mathcal{X}, \mathcal{A}, \mu, V)$  be a  $k$ -player free game. In what follows, we will think of  $x \in \mathcal{X}^n$  as  $n \times k$  matrices, where the  $i$ th row indicates the inputs of all  $k$  players in the  $i$ th coordinate, and the  $j$ th column indicates the inputs of the  $j$ th player. Thus  $x(i, \cdot)$  indicates the  $i$ th row of  $x$ , and  $x(\cdot, j)$  indicates the  $j$ th column. When we write  $x_S$  for some subset  $S \subseteq [n]$ , we mean the submatrix of  $x$  consisting of the rows indexed by  $i \in S$ .

Let  $X$  be an  $n \times k$ -partite register. Then we will also format  $X$  as a  $n \times k$  matrix, so  $X_{(i, \cdot)}$  and  $X_{(\cdot, j)}$  have the natural meaning. For a subset  $S \subseteq [n]$ ,  $X_S$  denotes the registers corresponding to the rows of  $X$  indexed by  $S$ . For an index  $j$ ,  $X_{(S, j)}$  denotes the  $j$ th column of the rows indexed by  $S$ .  $X_{(S, -j)}$  denotes the submatrix of  $X$  corresponding to rows indexed by  $i \in S$ , and all columns except for the  $j$ th one.



We make the following observation, which will be useful for us in our analysis: without loss of generality, we can restrict our attention to free games whose input distribution is the uniform distribution over some alphabet. Let  $G = (\mathcal{X}, \mathcal{A}, \mu, V)$  be a  $k$ -player free game. Write  $\mu = \mu_1 \otimes \cdots \otimes \mu_k$ , where  $\mu_j$  is a distribution over the alphabet  $\mathcal{X}_j$ . Fix an  $\gamma > 0$ . For each  $i$ , there exists an alphabet  $\mathcal{X}'_j$  and a map  $f_j : \mathcal{X}'_j \rightarrow \mathcal{X}_j$  such that the random variable  $X'_j = f_j(U_j)$  (where  $U_j$  is a uniformly random element from  $\mathcal{X}'_j$ ) is  $\gamma/k$ -close in total variation distance to being distributed according to  $\mu_j$  – and hence the distribution of  $(f_1(U_1), \dots, f_k(U_k))$  is at most  $\gamma$ -far from  $\mu$ . Thus, we can “simulate” the game  $G$  with another game  $G' = (\mathcal{X}', \mathcal{A}, U, V')$ , where  $\mathcal{X}' = \mathcal{X}'_1 \times \cdots \times \mathcal{X}'_k$ ,  $U$  is the uniform distribution on  $\mathcal{X}'$ , and  $V' : \mathcal{X}' \times \mathcal{A} \rightarrow \{0, 1\}$  is the map  $(x', a) \rightarrow V(\langle f_1(x'_1), \dots, f_k(x'_k) \rangle, a)$ .

► **Claim 5.1.**  $\text{val}^*(G') = \text{val}^*(G) \pm \gamma$ .

**Proof.** Consider the optimal strategy for  $G$ . Then a strategy for  $G'$  is the following: player  $j$ , on input  $u'_j \in \mathcal{X}'_j$ , computes  $u_j = f_j(u'_j)$ , and performs the strategy she would've done for  $G$ . The input distribution, from the point of view of the strategy for  $G$ , is at most  $\gamma$ -far from the original input distribution  $\mu$ . Thus the winning probability is at least  $\text{val}^*(G) - \gamma$ .

Now consider the optimal strategy for  $G'$ . Then a strategy for  $G$  is the following: player  $j$ , on input  $u_j \in \mathcal{X}_j$ , computes a uniformly random preimage  $u'_j \in f_j^{-1}(u_j)$ , and performs the strategy she would've done for  $G'$ . The input distribution, from the point of view of the strategy for  $G'$ , is at most  $\gamma$ -far from the uniform distribution  $U$ . Thus the winning probability is at least  $\text{val}^*(G') - \gamma$ . ◀

Furthermore, this simulation “commutes” with parallel repetition, in that  $\text{val}^*((G')^{\otimes n}) = \text{val}^*(G^{\otimes n}) \pm \gamma n$ . We can make  $\gamma$  arbitrarily small, at the cost of (potentially) increasing the input alphabet size, so that the behavior of the simulation  $G'$  is essentially the same as the original game  $G$ . However, since our theorems do not depend on the input alphabet size, we will treat  $\gamma$  as infinitesimally small, and hence neglect it.

► **Theorem 5.2.** *Let  $G = (\mathcal{X}, \mathcal{A}, \mu, V)$  be a  $k$ -player free game with classical outputs and classical verification predicate  $V : \mathcal{X} \times \mathcal{A} \rightarrow \{0, 1\}$ . Suppose that  $\text{val}^*(G) = 1 - \epsilon$ . Let  $s = \max_j \log |\mathcal{A}_j|$ . Then, for all  $n > k^4 s \log(k^2/\epsilon)/\epsilon^{3/2}$ , we have that*

$$\text{val}^*(G^{\otimes n}) \leq (1 - \epsilon^{3/2})^{\Omega(n/k^4 s)}.$$

**Proof.** Because of Claim 5.1, it is without loss of generality to assume that the input distribution  $\mu$  is the uniform distribution – the following analysis can be performed on a simulation of  $G$ , which will still bound the repeated game value of  $G$ .

Let  $n$  be some integer greater than  $k^4 s \log(k^2/\epsilon)/\epsilon^{3/2}$ , and consider an optimal entangled strategy for  $G^{\otimes n}$ , and let  $2^{-t}$  denote its winning probability. Suppose for contradiction that  $t \leq c\epsilon^{3/2}n/(k^4 s)$  for some universal constant  $c$ . Using this strategy, we will construct the following state

$$|\varphi\rangle^{XX'EA} := \sum_{x \in \mathcal{X}^n} \sqrt{\nu(x)} |xx\rangle^{XX'} \otimes |\varphi_x\rangle^{EA},$$

where  $\nu(x)$  is a probability distribution over  $\mathcal{X}^n$ ,  $X, X', A$  are  $n \times k$ -partite registers, and  $E$  is a  $k$ -partite register. We will show that exists a coordinate  $i \in [n]$ , and  $\delta < \epsilon/32k^2$  satisfying the following properties:

1. Measuring the  $X_{(i,\cdot)}A_{(i,\cdot)}$  register of  $\varphi$  yields a tuple  $(x_{(i,\cdot)}, a_{(i,\cdot)})$  that satisfies  $V(x_{(i,\cdot)}, a_{(i,\cdot)}) = 1$  with probability at least  $1 - \epsilon/8$ ;
2.  $S(\varphi^{X_{(i,\cdot)}} || \mu) \leq \delta$ .
3. For all  $j \in [k]$ ,  $I(X_{(i,j)} : Z_{-j})_\varphi \leq \delta$ , where  $Z_{-j} = X'_{(\cdot,-j)}X_{(\cdot,-j)}E_{-j}A_{(\cdot,-j)}$ .

For now, we assume the existence of such a state  $|\varphi\rangle$ ; we will construct it in Lemma 5.3. We use Lemma 4.3 on the state  $\varphi$  to obtain for each player  $j$  a set of unitaries  $\{U_u^j\}_{u \in \mathcal{X}_j}$  acting on  $X_{(\cdot,j)} X'_{(\cdot,j)} E_j A_{(\cdot,j)}$  such that

$$\mathbb{E}_{x_{(i,\cdot)} \leftarrow \varphi^{X_{(i,\cdot)}}} [K(\varphi_{x_{(i,\cdot)}}, \mathcal{U}_{x_{(i,\cdot)}}(\varphi))] \leq 4k \sum_j I(X_{(i,j)} : Z_{-j})_\varphi \leq 4k^2 \delta,$$

where we let  $U_{x_{(i,\cdot)}} = \bigotimes_j U_{x_{(i,j)}}^j$ , and let  $\mathcal{U}_{x_{(i,\cdot)}}$  be the CP map  $\varphi \mapsto U_{x_{(i,\cdot)}} \varphi U_{x_{(i,\cdot)}}^\dagger$ . The state  $\varphi_{x_{(i,\cdot)}}$  denotes  $\varphi$  conditioned on  $X_{(i,\cdot)} = x_{(i,\cdot)}$ .

We now describe a protocol for the  $k$  players to play game  $G$ . The players receive  $u \in \mathcal{X}$ , drawn from the product distribution  $\mu$ . Player  $j$  receives  $u_j \in \mathcal{X}_j$ . The players share the state  $\varphi$ , where player  $j$  has access to the  $X_{(\cdot,j)} X'_{(\cdot,j)} E_j A_{(\cdot,j)}$  registers.

### Protocol A

**Input:**  $u \in \mathcal{X}$ . Player  $j$  receives  $u_j$ .

**Preshared entanglement:**  $\varphi$

**Strategy for player  $j$ :**

1. Apply the local unitary  $U_{u_j}^j$  on the  $X_{(\cdot,j)} X'_{(\cdot,j)} E_j A_{(\cdot,j)}$  registers of  $\varphi$ .
2. Output the  $A_{(i,j)}$  part of  $\varphi$ .

Slightly overloading notation, we let  $V_u^i$  denote the projector  $\sum_{a \in \mathcal{A}: V(u,a)=1} |a\rangle\langle a|$  that acts on the  $A_{(i,\cdot)}$  registers. Let  $\kappa$  denote the winning probability of Protocol A. This is equal to

$$\begin{aligned} \kappa &= \mathbb{E}_{u \leftarrow \mu} \|V_u^i U_u |\varphi\rangle\|^2 \\ &\geq \mathbb{E}_{u \leftarrow \varphi^{X_{(i,\cdot)}}} \|V_u^i U_u |\varphi\rangle\|^2 - 4\delta. \end{aligned}$$

where we use property **(B)** and appeal to Lemma 3.15. Let

$$\tau := \mathbb{E}_{u \leftarrow \varphi^{X_{(i,\cdot)}}} \|V_u^i U_u |\varphi\rangle\|^2.$$

For every  $i \in [n]$ ,  $u \in \mathcal{X}$ , define the quantum operation  $\mathcal{E}_{i,u}$  that, given a state  $\varphi$ , measures the  $A_{(i,\cdot)}$  registers using  $V_u^i$  measurement, and outputs a classical binary random variable  $F$  indicating the verification measurement outcome (outcome 1 corresponds to “accept” and outcome 0 corresponds to “reject”). Let

$$F_0 = \mathbb{E}_{x_{(i,\cdot)} \leftarrow \varphi^{X_{(i,\cdot)}}} \mathcal{E}_{i,x_{(i,\cdot)}}(\varphi_{x_{(i,\cdot)}}) \quad \text{and} \quad F_1 = \mathbb{E}_{u \leftarrow \varphi^{X_{(i,\cdot)}}} \mathcal{E}_{i,u}(\mathcal{U}_u(\varphi)).$$

Note that  $\Pr(F_0 = 1) \geq 1 - \epsilon/8$  by our assumption on  $\varphi$ , and  $\Pr(F_1 = 1) = \tau$ . Then,

$$\begin{aligned} K(F_0, F_1) &= K\left(\mathbb{E}_{x_{(i,\cdot)} \leftarrow \varphi^{X_{(i,\cdot)}}} \mathcal{E}_{i,x_{(i,\cdot)}}(\varphi_{x_{(i,\cdot)}}), \mathbb{E}_{u \leftarrow \varphi^{X_{(i,\cdot)}}} \mathcal{E}_{i,u}(\mathcal{U}_u(\varphi))\right) \\ &\leq \mathbb{E}_{x_{(i,\cdot)} \leftarrow \varphi^{X_{(i,\cdot)}}} K(\mathcal{E}_{i,x_{(i,\cdot)}}(\varphi_{x_{(i,\cdot)}}), \mathcal{E}_{i,x_{(i,\cdot)}}(\mathcal{U}_{x_{(i,\cdot)}}(\varphi))) \quad (\text{Fact 3.4}) \\ &\leq \mathbb{E}_{x_{(i,\cdot)} \leftarrow \varphi^{X_{(i,\cdot)}}} K(\varphi_{x_{(i,\cdot)}}, \mathcal{U}_{x_{(i,\cdot)}}(\varphi)) \quad (\text{Fact 3.2}) \\ &\leq 4k^2 \delta. \end{aligned}$$

By our assumption on  $\delta$ , this is at most  $K(F_0, F_1) \leq \epsilon/8$ . By Lemma 3.16,  $\Pr(F_1 = 1) \geq 1 - \epsilon/8 - \epsilon/2$ . Thus  $\kappa \geq 1 - 3\epsilon/4$ . But notice that Protocol A is a valid strategy for the game  $G$ ; thus we have produced a strategy for game  $G$  that wins with probability strictly greater than  $1 - \epsilon$ , a contradiction. Thus, it must be at  $t = \Omega(\epsilon^{3/2}n/(k^4s))$ , which establishes the theorem.  $\blacktriangleleft$

### 5.1 Construction of $\varphi$

► **Lemma 5.3.** *There exists a state  $|\varphi\rangle$ , and a coordinate  $i \in [n]$  satisfying properties (A), (B), and (C).*

**Proof.** Set  $\epsilon' = \epsilon/32$ ,  $\eta = 2^{-t}\epsilon/32k^2$ , and  $h = c' \log(1/\eta)/\epsilon'$  for some constant  $c'$ . We have  $h = (32c'/\epsilon)(t + \log(32k^2/\epsilon))$ , and by our assumptions on  $t$  and  $n$ , this is at most  $n/2$ .

Suppose there was a strategy to win the repeated game  $G^{\otimes n}$  with probability  $2^{-t}$ , involving a shared state  $|\xi\rangle^E$  (where  $E$  is a  $k$ -partite state register) and measurements  $\{M_a^{j,x_j}\}$  for the players, respectively. That is, player  $j$ , on input  $x_j \in \mathcal{X}_j^n$ , applies the measurement with POVM elements  $\{M_a^{j,x_j}\}$  and reports the outcome.

We will build the state  $\varphi$  in steps. Consider the initial state

$$|\psi^0\rangle := \sum_{x \in \mathcal{X}^n} \sqrt{\mu^{\otimes n}(x)} |xx\rangle^{XX'} \otimes \sum_{a \in \mathcal{A}^n} |\xi_{xa}\rangle^E \otimes |a\rangle^A$$

where  $|\xi_{xa}\rangle = \left(\bigotimes_j \sqrt{M_{a_j}^{j,x_j}}\right) |\xi\rangle$  (which is a subnormalized state), and  $\mu^{\otimes n}(x)$  is the probability distribution associated with the repeated game  $G^{\otimes n}$ . For every set  $C \subset [n]$ , and every fixing of the inputs  $x_C$  to the coordinates indexed by  $C$ , define the state  $|\psi_{C,x_C}^0\rangle$  to be  $|\psi^0\rangle$  conditioned on  $X_C = x_C$ .

Now consider the following  $k$ -player communication protocol: for every set  $C \subset [n]$  and every  $x_C$ , the players share the entangled state  $|\psi_{C,x_C}^0\rangle$ , where player  $j$  has access to the registers  $X_{(\cdot,j)} X'_{(\cdot,j)} E_j A_{(\cdot,j)}$ . Using shared randomness, the players sample  $h$  independent and uniformly random coordinates  $C = \{i_1, \dots, i_h\} \subset [n]$ , and sample  $x_C$  from the marginal distribution of  $\mu^{\otimes n}$  on the subset  $C$ . For the remainder of the protocol, the players perform all their operations on the shared state  $|\psi_{C,x_C}^0\rangle$ .

In the next phase of the protocol, the  $k$  players communicate qubits to each other to determine whether they have won or lost the parallel repeated game  $G^{\otimes n}$ . In particular, they run a protocol to search for a coordinate  $i \in C$  such that  $V(x_{(i,\cdot)}, a_{(i,\cdot)}) = 0$ , if it exists – call such a coordinate a losing coordinate. The state  $|\psi_{C,x_C}^0\rangle$  becomes transformed to

$$|\psi_{C,x_C}\rangle^{XX'EAR} := \sum_{x \in \mathcal{X}^n} \sqrt{\mu^{\otimes n}(x|x_C)} |xx\rangle^{XX'} \otimes \sum_{a \in \mathcal{A}^n} |\xi'_{Cxa}\rangle^E \otimes |a\rangle^A \otimes (\alpha_{Cxa} |1\rangle + \beta_{Cxa} |0\rangle)^R$$

where  $\mu^{\otimes n}(x|x_C)$  is probability of  $x$  conditioned on  $x_C$ , and  $|\xi'_{Cxa}\rangle = |\xi_{Cxa}\rangle \otimes |w_{Cxa}\rangle$  with  $|w_{Cxa}\rangle$  denoting the workspace qubits that are used during the protocol. The coefficients  $\alpha_{Cxa}$  and  $\beta_{Cxa}$  denote the amplitude that the search protocol places on the flags “No losing coordinates” and “Exists a losing coordinate” respectively.

For now, we will abstract away from the particulars of this communication protocol and defer the details of it until later. The only things we will use about this search protocol is the following:

1. The search protocol is run conditioned on  $C$ , and the  $XA$  registers;
2. At most  $T = O(\sqrt{1/\epsilon'} \log(1/\eta) \log |\mathcal{A}|)$  qubits in total are exchanged between all parties, where  $\mathcal{A}$  is the output alphabet in game  $G$ ;

3. For every fixing of  $(x, a) \in \mathcal{X}^n \times \mathcal{A}^n$ , if there are no coordinates  $i \in [n]$  such that  $V(x_{(i,\cdot)}, a_{(i,\cdot)}) = 0$ , then the search procedure reports “No losing coordinates” with certainty; and
4. If there are at least an  $\epsilon'n$  bad coordinates, then the search procedure reports “No losing coordinates” with probability at most  $\eta$  (over the quantum randomness of the protocol, as well as over the choice of  $C$ ). In other words, for tuples  $(x, a) \in \mathcal{X}^n \times \mathcal{A}^n$  with  $\mathbb{E}_i[V(x_{(i,\cdot)}, a_{(i,\cdot)})] < 1 - \epsilon'$ ,

$$\sum_C p(C) |\alpha_{Cxa}|^2 \leq \eta,$$

where  $p(C)$  is the distribution that samples  $h$  independent and uniformly random coordinates from  $[n]$ .

For all  $C, x_C$  define  $|\varphi_{C,x_C}\rangle$  to be  $|\psi_{C,x_C}\rangle$  conditioned on measuring 1 in the  $R$  register:

$$|\varphi_{C,x_C}\rangle^{XX'EAR} := \frac{1}{\sqrt{\lambda_{C,x_C}}} \sum_{x \in \mathcal{X}^n} \sqrt{\mu^{\otimes n}(x|x_C)} |xx\rangle^{XX'} \otimes \sum_{a \in \mathcal{A}^n} |\xi'_{Cxa}\rangle^E \otimes |a\rangle^A \otimes (\alpha_{Cxa} |1\rangle^R)$$

where  $\lambda_{C,x_C}$  is for normalization. In the case that  $\lambda_{C,x_C} = 0$  (meaning that we were trying to normalize the 0 state), we leave the state undefined. Let  $\psi^{CX_C}(C, x_C) = p(C)\mu^C(x_C)$  denote the joint probability distribution of the shared random variables  $C$  and  $X_C$ , before conditioning. Let  $\varphi^{CX_C}(C, x_C) = p(C)\mu^C(x_C)\lambda_{C,x_C}/\lambda$  denote the joint distribution conditioned on  $R = 1$ , where  $\lambda = \sum_{C,x_C} \psi^{CX_C}(C, x_C)\lambda_{C,x_C}$ .

**(A) A random coordinate of  $\varphi$  wins with high probability.** Let

$$\rho = \mathbb{E}_{C,x_C \leftarrow \psi^{CX_C}} [ |C\rangle\langle C| \otimes |x_C\rangle\langle x_C| \otimes \psi_{C,x_C} ]$$

and

$$\sigma = \mathbb{E}_{C,x_C \leftarrow \varphi^{CX_C}} [ |C\rangle\langle C| \otimes |x_C\rangle\langle x_C| \otimes \varphi_{C,x_C} ].$$

Observe that  $\sigma$  is the post-measurement state of  $\rho$  after measuring  $|1\rangle$  in the  $R$  register. Let  $\mathcal{E}$  denote the quantum operation on that, (1) measures the  $C$  register, (2) chooses a uniformly random  $i \notin C$ , (3) measures  $X_{(i,\cdot)}$  register, and (4) then conditioned on  $X_{(i,\cdot)} = x_{(i,\cdot)}$ , performs the binary verification measurement  $V_{x_{(i,\cdot)}}^i$  defined in the previous section, setting an auxiliary register  $Q$  to  $|1\rangle$  if the measurement accepts,  $|0\rangle$  if it rejects. We wish to argue that the probability that a measurement of the  $Q$  register of  $\mathcal{E}(\sigma)$  yields 1 with high probability. This probability is equivalent to the probability the following process succeeds: first, measure the  $XA$  registers of  $\sigma$  to obtain a tuple  $(x, a)$ . Then, measure the  $C$  register. Finally, select a random index  $i \notin C$ , and we succeed if  $V(x_{(i,\cdot)}, a_{(i,\cdot)}) = 1$ .

In this alternative process, the probability that we measure  $(x, a)$  in  $\sigma$  such that  $\mathbb{E}_{i \in [n]}[V(x_{(i,\cdot)}, a_{(i,\cdot)})] < 1 - \epsilon'$  (call such  $(x, a)$ 's “bad”) is equal to

$$\frac{1}{\lambda} \sum_{(x,a) \text{ bad}} \Pr_{\rho}(x, a) \sum_C p(C) |\alpha_{Cxa}|^2$$

where  $\Pr_{\rho}(x, a)$  is the probability of measuring measuring  $(x, a)$  in  $\rho$ . By our assumption on the communication protocol, this is at most  $\eta/\lambda$ . Since the players' strategy wins the repeated game  $G^{\otimes n}$  with probability  $2^{-t}$ , we have that  $\lambda \geq 2^{-t}$ . Thus the probability of measuring a bad  $(x, a)$  is at most  $2^t \eta$ .

Now suppose we measure  $(x, a)$  such that  $\mathbb{E}_{i \in [n]}[V(x_{(i,\cdot)}, a_{(i,\cdot)})] \geq 1 - \epsilon'$ . Then, for any  $C$ , a random  $i \notin C$  loses with probability at most  $\epsilon'n/(n - |C|) \leq \epsilon'n/(n - h) \leq \epsilon/16$ . Thus, the probability that the  $Q$  register of  $\mathcal{E}(\sigma)$  yields 0 is at most  $2^t \eta + \epsilon/16$ .

**(B) Coordinate input distributions are mostly unaffected.** By Fact 3.11, since  $\sigma \preceq 2^\lambda \rho$ , we have

$$\begin{aligned} \log 1/\lambda &\geq S_\infty(\sigma\|\rho) \\ &\geq S(\sigma\|\rho) \\ &\geq \mathbb{E}_{C, x_C \leftarrow \varphi^{CX_C}} S(\varphi_{C, x_C}^{XX'EA} \|\psi_{C, x_C}^{XX'EA}), \end{aligned} \quad (5.1)$$

where in the last line we used Fact 3.10. Using Facts 3.8 and 3.9, we obtain that

$$\begin{aligned} \log 1/\lambda &\geq \mathbb{E}_{C, x_C \leftarrow \varphi^{CX_C}} S(\varphi_{C, x_C}^X \|\psi_{C, x_C}^X) \\ &\geq \mathbb{E}_{C, x_C \leftarrow \varphi^{CX_C}} \sum_{i \notin C} S(\varphi_{C, x_C}^{X(i, \cdot)} \|\psi_{C, x_C}^{X(i, \cdot)}) \\ &= \mathbb{E}_{C, x_C \leftarrow \varphi^{CX_C}} \sum_{i \notin C} S(\varphi_{C, x_C}^{X(i, \cdot)} \|\mu). \end{aligned}$$

**(C) Mutual information is small.**

► **Claim 5.4.** Fix a  $j \in [k]$ , and fix a  $C, x_C$ . There exists a state  $\sigma_{C, x_C}^{Z-j}$  such that

$$S_\infty(\psi_{C, x_C}^{X(\cdot, j)Z-j} \|\psi_{C, x_C}^{X(\cdot, j)} \otimes \sigma_{C, x_C}^{Z-j}) \leq 2T,$$

where  $Z_{-j} = X_{(\cdot, -j)} X'_{(\cdot, -j)} E_{-j} A_{(\cdot, -j)}$ .

We defer the proof of this claim for later, and will assume it for now. Line (5.1) with Fact 3.8 implies that for all  $j$ ,  $\mathbb{E}_{C, x_C \leftarrow \varphi^{CX_C}} S(\varphi_{C, x_C}^{X(\cdot, j)Z-j} \|\psi_{C, x_C}^{X(\cdot, j)Z-j}) \leq \log 1/\lambda$ . Using Fact 3.14 with Claim 5.4, we get that for all  $j$ , there exists a  $\sigma_{C, x_C}^{Z-j}$  such that

$$\mathbb{E}_{C, x_C \leftarrow \varphi^{CX_C}} S(\varphi_{C, x_C}^{X(\cdot, j)Z-j} \|\psi_{C, x_C}^{X(\cdot, j)} \otimes \sigma_{C, x_C}^{Z-j}) \leq 2T + \log 1/\lambda.$$

Using Quantum Raz's Lemma, we get

$$\mathbb{E}_{C, x_C \leftarrow \varphi^{CX_C}} \mathbb{E}_{i \in [n]} I(X_{(i, j)} : Z_{-j})_{\varphi_{C, x_C}} \leq 2(\log 1/\lambda + 2T)/n.$$

By Markov's inequality, we have that there exists a  $C, x_C, i \notin C$  such that

1. Measuring the  $X_{(i, \cdot)} A_{(i, \cdot)}$  register of  $\varphi_{C, x_C}$  yields a tuple  $(x_{(i, \cdot)}, a_{(i, \cdot)})$  that satisfies  $V(x_{(i, \cdot)}, a_{(i, \cdot)}) = 1$  with probability at least  $1 - \epsilon/8$ .
2.  $S(\varphi_{C, x_C}^{X(i, \cdot)} \|\mu) \leq 32t/n$ .
3. For all  $j \in [k]$ ,  $I(X_{(i, j)} : Z_{-j})_{\varphi_{C, x_C}} \leq 64k(t + 2T)/n$ .

Let  $\delta = 64k(t + 2T)/n$ . Let  $s$  be the maximum number of qubits output by any one player in game  $G$ , so  $ks \geq \log |\mathcal{A}|$ . Then the total communication is  $T = O(ks(t + \log k^2/\epsilon)/\sqrt{\epsilon})$ . Then, if  $t \leq c\epsilon^{3/2}n/(k^4s)$  for some universal constant  $c$ , we have  $\delta \leq \epsilon/32k^2$ . Let  $\varphi = \varphi_{C, x_C}$ . This yields the state and coordinate  $i$  required. ◀

## 5.2 The search protocol

Next, we detail the search protocol used to construct  $|\psi\rangle$  and  $|\varphi\rangle$ . We describe the protocol for a two-player game  $G$ ; the extension to  $k$  parties is straightforward.

Let  $G = (\mathcal{X} \times \mathcal{Y}, \mathcal{A} \times \mathcal{B}, \mu, V)$  be a two-player free game, where  $\mathcal{X}$  and  $\mathcal{Y}$  are Alice and Bob's input alphabets, respectively, and  $\mathcal{A}$  and  $\mathcal{B}$  are their output alphabets. Consider the optimal strategy for  $G^{\otimes n}$ , where there is a shared state  $|\xi\rangle^{E_A E_B}$  where on input  $(x, y) \in \mathcal{X}^n \times \mathcal{Y}^n$ , Alice and Bob apply measurements  $\{M_a^x\}_{a \in \mathcal{A}^n}$  and  $\{N_b^y\}_{b \in \mathcal{B}^n}$  respectively on their share of  $|\xi\rangle$ .

At the start of the search protocol, a multiset  $C = \{i_1, \dots, i_h\}$ ,  $x_C \in \mathcal{X}^C$ , and  $y_C \in \mathcal{Y}^C$  are publically visible to Alice and Bob. They are both given the state

$$|\psi_{C, x_C, y_C}^0\rangle = \sum_{x \in \mathcal{X}^n, y \in \mathcal{Y}^n} \sqrt{\mu^{\otimes n}(x, y | x_C, y_C)} |x y y\rangle^{X X' Y Y'} |\xi\rangle^{E_A E_B} |0\rangle^R$$

where  $\mu^{\otimes n}(x, y | x_C, y_C)$  is the distribution of  $(x, y)$  conditioned on  $x_C, y_C$ . Alice has access to registers  $X X' E_A R$ , and Bob has access to registers  $E_B Y Y'$ .

Then, Alice and Bob apply their measurements from the optimal strategy, controlled on the  $X$  and  $Y$  registers, respectively, to obtain

$$|\psi_{C, x_C, y_C}^1\rangle = \sum_{x \in \mathcal{X}^n, y \in \mathcal{Y}^n} \sqrt{\mu^{\otimes n}(x, y | x_C, y_C)} |x y y\rangle \sum_{a \in \mathcal{A}^n, b \in \mathcal{B}^n} |\xi_{xyab}\rangle |ab\rangle |0\rangle$$

where  $|\xi_{xyab}\rangle = (\sqrt{M_a^x} \otimes \sqrt{N_b^y}) |\xi\rangle$ .

Alice and Bob then run a distributed search protocol controlled on the  $XYAB$  registers. Fix  $(x, y, a, b)$ . The protocol proceeds as follows: Alice and Bob divide the multiset  $C$  into groups  $D_1, \dots, D_q$ , each of size  $m = \lceil 1/\epsilon' \rceil$ . For each  $\ell = 1, \dots, q$ , Alice and Bob perform a distributed version of the Aaronson-Ambainis 3-dimensional search algorithm [1] to determine whether there is a coordinate  $D_\ell$  contains a losing coordinate – i.e., a coordinate  $i \in D_\ell$  such that  $V(x_i, y_i, a_i, b_i) = 0$ .

The search protocol for a group  $D_\ell$  works as follows. Whenever the Aaronson-Ambainis algorithm is in the state  $\sum_i \gamma_{i,z} |i, z\rangle$ , where  $|i\rangle$  corresponds to an index in  $D_\ell$ , and  $|z\rangle$  is a qubit indicating whether a marked item has been found, the joint state between Alice and Bob will be  $\sum_i \gamma_{i,z} |i\rangle \otimes |z\rangle \otimes |i\rangle$ , where Alice holds the first  $|i\rangle$  and  $|z\rangle$ , and Bob holds the second  $|i\rangle$ . Thus, Alice and Bob query locations are “synchronized”. When Aaronson-Ambainis algorithm has to perform a query controlled on  $|i\rangle$ , Bob sends the qubit containing  $|b_i\rangle$ . Alice, controlled on  $|b_i\rangle$ , performs  $|z\rangle \mapsto |z \oplus V(x_i, y_i, a_i, b_i) \oplus 1\rangle$  – note that Alice can perform this, because in addition to  $x_i, a_i$ , and  $b_i$ , she also has access to  $y_i$  because  $y_C$  is public. We perform an additional XOR with 1 because a “marked item” for the search algorithm corresponds to a *losing* coordinate. Alice then sends back  $|b_i\rangle$  to Bob. The other non-query transformations of the Aaronson-Ambainis algorithm are handled as in the the protocol described in [1]. Each step of the algorithm incurs at most  $O(\log |\mathcal{B}|)$  qubits of communication, and there are  $O(\sqrt{m})$  steps, resulting in  $O(\sqrt{m} \log |\mathcal{B}|)$  qubits of total communication. If  $D_\ell$  contains a losing coordinate, then this protocol will succeed in finding one with probability at least  $2/3$ .

If for at least one  $\ell$ , Alice and Bob find a losing coordinate for  $G_\ell$ , Alice sets the  $R$  register to 0; otherwise, it sets it to 1. Thus the total amount of communication of this protocol is  $T = O(q\sqrt{m} \log |\mathcal{B}|) = O(\sqrt{1/\epsilon'} \log 1/\eta \log |\mathcal{B}|)$ . The final state of the protocol looks like

$$|\psi_{C, x_C, y_C}\rangle = \sum_{x, y} \sqrt{\mu^{\otimes n}(x, y | x_C, y_C)} |x y y\rangle \sum_{a, b} |\xi'_{xyab}\rangle |ab\rangle (\alpha_{Cxyab} |1\rangle + \beta_{Cxyab} |0\rangle),$$

where  $|\xi'_{xyab}\rangle = |\xi_{xyab}\rangle \otimes |w_{Cxyab}\rangle$  with  $|w_{Cxyab}\rangle$  denoting the workspace qubits of the two players.

Fix a setting of the registers  $XYAB = (x, y, a, b)$ . Suppose there was no  $i \in [n]$  such that  $V(x_i, y_i, a_i, b_i) = 0$ . Then the search algorithm will never find a losing coordinate in any of the  $G_\ell$ 's, so for all  $C$ , we have  $\beta_{Cxyab} = 0$ . On the other hand, suppose there were at least  $\epsilon'n$  losing coordinates. We analyze, for a fixed  $(x, y, a, b)$ , the error quantity  $\sum_C p(C) |\alpha_{Cxyab}|^2$ . We can write  $p(C) = \prod_\ell p(D_\ell)$ , because each index in  $C$  is chosen uniformly and independently at random. Furthermore, we can decompose  $|\alpha_{Cxyab}|^2 = \prod_\ell |\alpha_{D_\ell xyab}|^2$ , where  $\alpha_{D_\ell xyab}$  is the probability amplitude that the Aaronson-Ambainis protocol does not find a losing coordinate in  $D_\ell$ . Thus the error quantity can be written as  $\prod_\ell \sum_{D_\ell} p(D_\ell) |\alpha_{D_\ell xyab}|^2 = (\sum_D p(D) |\alpha_{Dxyab}|^2)^q$ . Each  $D_\ell$  independently has at least  $1 - (1 - \epsilon')^m \geq 1 - 1/e$  probability of containing a losing coordinate. When  $D_\ell$  has a losing coordinate, the Aaronson-Ambainis search protocol will succeed in finding it with probability at least  $2/3$ . Thus the error quantity is at most

$$\begin{aligned} & (\Pr(D \text{ contains losing coordinate}) \cdot (1/3) + \Pr(D \text{ does not have losing coordinate}))^q \\ & \leq (1/3 + 1/e)^q \\ & = \exp(-\Omega(q)) = \eta. \end{aligned}$$

This establishes the requisite properties of the search protocol in the case of  $k = 2$ .

The extension to general  $k$  parties is straightforward. At the beginning of the protocol, a multiset  $C = D_1 \cdots D_q$  and inputs  $x_C$  are publically visible to all players. They start with an analogous initial state  $|\psi_{C, x_C}\rangle$ , where each player  $j$  has access to registers  $X_{(\cdot, j)} X'_{(\cdot, j)} E_j A_{(\cdot, j)}$ ; player 1 also has access to register  $R$ . They perform the distributed Aaronson-Ambainis protocol independently on all  $D_\ell$ . There are  $k - 1$  communication channels, one between the first player and all the other players. Whenever a query is to be made, player  $j \in \{2, \dots, k\}$  sends her answer symbol  $a_{(i, j)}$  the first player, who then computes  $V(x_{(i, \cdot)}, a_{(i, \cdot)})$ . The other non-query transformations of the algorithm are also easily extended to the multiplayer case. The total communication is  $T = O(q\sqrt{m} \log |\mathcal{A}|) = O(\sqrt{1/\epsilon'} \log 1/\eta \log |\mathcal{A}|)$ , where  $\mathcal{A}$  is the output alphabet for all  $k$  players.

**Proof of Claim 5.4.** Fix a  $C, x_C$ . Fix a player  $j \in [k]$ . Take the start state  $\psi_{C, x_C}^0$  defined above (extended appropriately to  $k$  players), and trace out the  $X'_{(\bar{C}, j)}$  register:  $\theta_{C, x_C}^0 = \text{tr}_{X'_{(\bar{C}, j)}}(\psi_{C, x_C}^0)$ . Since  $\mu^{\otimes n}$  is a product distribution across players and also across game coordinates, we have that  $\theta_{C, x_C}^0 = U^{X_{(\bar{C}, j)}} \otimes \phi_{C, x_C}^0$  where  $U^{X_{(\bar{C}, j)}}$  is the maximally mixed state for the register  $X_{(\bar{C}, j)}$ , and

$$|\phi_{C, x_C}^0\rangle = |x_C\rangle^{X_C X'_C} \sum_{x_{(\cdot, -j)}} \sqrt{\mu_{-j}^{\otimes n}(x_{(\cdot, -j)} | x_C)} |x_{(\cdot, -j)} x_{(\cdot, -j)}\rangle^{X_{(\bar{C}, -j)} X'_{(\bar{C}, -j)}} |\xi\rangle^E |0\rangle^R$$

where  $\mu_{-j}^{\otimes n}$  denotes the marginal distribution of  $\mu^{\otimes n}$  on all players inputs, except for the  $j$ th player. Here, we used the simplifying assumption that  $\mu$  is the uniform distribution. The search protocol described above never interacts with the  $X'_{\bar{C}}$  registers. Thus, we can view the protocol as the  $j$ th player receiving a uniformly random input drawn from,  $U^{X_{(\bar{C}, j)}}$ , and shares an entangled state  $\phi_{C, x_C}^0$  with players  $[k] - \{j\}$ . The rest of the protocol is some two-way communication between player  $j$  and every one else.  $\blacktriangleleft$

We now wish to analyze the min-entropy of player  $j$ 's input register  $X_{\bar{C}, j}$  relative to the state of all other players. We appeal to the beautiful result of Nayak and Salzman [17], whose theorem statement we reproduce here:



► **Theorem 5.5** ([17]). *Consider a communication protocol, without prior entanglement, where Alice receives a uniformly random  $n$ -bit input  $X$ , and interacts with Bob over a quantum communication channel. Let  $\psi^{XB}$  be the final joint state of Alice’s input  $X$  and Bob’s state in the protocol. Then, for any measurement strategy  $\{M_x\}_x$  that Bob applies to his own state, the probability that Bob guesses Alice’s input  $X$  correct is at most  $2^{2m_A}/2^n$ , where  $m_A$  is the number of qubits sent from Alice to Bob over the course of the protocol.*

We now rephrase their theorem to use relative min-entropy instead of guessing probabilities. Let  $\alpha$  be the optimal guessing probability for Bob. Then, the *quantum conditional min-entropy*  $H_{\min}(X|B)_\psi$  is defined to be  $-\log \alpha$ . However, by SDP duality, we have the alternative characterization that  $H_{\min}(X|B)_\psi = -\inf_{\sigma^B} S_\infty(\psi^{XB} \parallel \text{id}^X \otimes \sigma^B)$  [16]. Let  $\sigma^B$  be a state achieving this infimum. Then  $\log \alpha = S_\infty(\psi^{XB} \parallel \text{id}^X \otimes \sigma^B) = S_\infty(\psi^{XB} \parallel \frac{1}{2^n} \text{id}^X \otimes \sigma^B) - n$ . By the theorem of Nayak and Salzman,  $\log \alpha \leq 2m_A - n$ , so  $S_\infty(\psi^{XB} \parallel \frac{1}{2^n} \text{id}^X \otimes \sigma^B) = S_\infty(\psi^{XB} \parallel \psi^X \otimes \sigma^B) \leq 2m_A$ , where we used the fact that  $\psi^X$  is the uniform distribution.

To apply this theorem to our setting, we can treat player  $j$  as “Alice” and the rest of the players as “Bob”. Alice exchanges at most  $T$  qubits with Bob. The crucial component of the Nayak-Salzman theorem is that Bob’s probability of guessing does not depend on how many qubits he sent to Alice! Thus, we can imagine that in the beginning of the protocol he sent the  $E_j$  register of the shared entangled state  $\phi_{C,x_C}^0$  to Alice first. We have that there exists a state  $\sigma_{C,x_C}^{Z-j}$  such that

$$2T \geq S_\infty(\psi_{C,x_C}^{X(\cdot,j)Z-j} \parallel \psi_{C,x_C}^{X(\cdot,j)} \otimes \sigma_{C,x_C}^{Z-j}).$$

## 6 Open problems

We conclude with a variety of open problems.

1. Is it possible to extend the Grover search analysis to handle CQ games?
2. Is *strong parallel repetition* possible with the entangled value of free games? In other words, can the base of  $1 - \epsilon^{3/2}$  of Theorem 1.1 be improved to  $1 - \epsilon$ ?
3. Is the base of  $1 - \epsilon^2$  for the repeated *classical* value of free games tight? If so, this would mean that there is a separation of classical and quantum parallel repetition for free games.
4. It was shown by [10] that the dependence on the output alphabet size, for classical parallel repetition, is necessary – even for free games. However, Holenstein showed the repeated game value for non-signaling games has no such alphabet dependence [12]. Is this dependence necessary for the quantum case?
5. Can we identify an interesting class of games for which we can prove improved parallel repetition theorems, by designing efficient communication protocols to generate advice states?
6. The mantra, “Better parallel repetition theorems from better communication protocols,” suggests an intriguing connection between games and communication protocols. Although games are protocols that forbid communication between the players, one can define the *communication complexity of a game* as the minimum communication needed for the players to determine whether they have won or lost the game. Our mantra suggests a relationship between the value and communication complexity of a game. What is the nature of this relationship?
7. Can one use these techniques to prove parallel repetition for entangled games with an arbitrary input distribution?

**Acknowledgments.** We thank Scott Aaronson for helpful comments about the Aaronson-Ambainis algorithm. We also thank Thomas Vidick and André Chailloux for helpful comments on earlier versions of this paper. Finally, we thank the anonymous referees for helpful feedback. HY is supported by an NSF Graduate Fellowship Grant No. 1122374 and National Science Foundation Grant No. 1218547. XW is funded by ARO contract W911NF-12-1-0486 and by the NSF Waterman Award of Scott Aaronson. Part of this research was conducted while XW was a Research Fellow, and HY a visiting graduate student, at the Simons Institute for the Theory of Computing, University of California, Berkeley.

---

## References

- 1 Scott Aaronson and Andris Ambainis. Quantum search of spatial regions. In *Foundations of Computer Science, 2003. Proceedings. 44th Annual IEEE Symposium on*, pages 200–209. IEEE, 2003.
- 2 Rotem Arnon-Friedman, Renato Renner, and Thomas Vidick. Non-signalling parallel repetition using de finetti reductions. *arXiv preprint arXiv:1411.1582*, 2014.
- 3 Boaz Barak, Anup Rao, Ran Raz, Ricky Rosen, and Ronen Shaltiel. Strong parallel repetition theorem for free projection games. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 352–365. Springer, 2009.
- 4 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*, pages 63–68. ACM, 1998.
- 5 Harry Buhrman, Serge Fehr, and Christian Schaffner. On the parallel repetition of multi-player games: The no-signaling case. *arXiv preprint arXiv:1312.7455*, 2013.
- 6 André Chailloux and Giannicola Scarpa. Parallel repetition of entangled games with exponential decay via the superposed information cost. 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 296–307. Springer, 2014.
- 7 André Chailloux and Giannicola Scarpa. Parallel repetition of free entangled games: Simplification and improvements. *arXiv preprint arXiv:1410.4397*, 2014.
- 8 Kai-Min Chung, Xiaodi Wu, and Henry Yuen. Parallel repetition for entangled k-player games via fast quantum search. *arXiv preprint arXiv:1501.00033*, 2015.
- 9 Irit Dimur, David Steurer, and Thomas Vidick. A parallel repetition theorem for entangled projection games. In *IEEE 29th Conference on Computational Complexity, CCC 2014, Vancouver, BC, Canada, June 11-13, 2014*, pages 197–208, 2014.
- 10 Uriel Feige and Oleg Verbitsky. Error reduction by parallel repetition: a negative result. *Combinatorica*, 22(4):461–478, 2002.
- 11 Joseph Fitzsimons and Thomas Vidick. A multiprover interactive proof system for the local hamiltonian problem. *arXiv preprint arXiv:1409.0260*, 2014.
- 12 Thomas Holenstein. Parallel repetition: simplifications and the no-signaling case. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing*, pages 411–419. ACM, 2007.
- 13 Rahul Jain, Attila Pereszlényi, and Penghui Yao. A parallel repetition theorem for entangled two-player one-round games under product distributions. In *IEEE 29th Conference on Computational Complexity, CCC 2014, Vancouver, BC, Canada, June 11-13, 2014*, pages 209–216, 2014.

- 14 Rahul Jain, Jaikumar Radhakrishnan, and Pranab Sen. A lower bound for the bounded round quantum communication complexity of set disjointness. In *Foundations of Computer Science, 2003. Proceedings. 44th Annual IEEE Symposium on*, pages 220–229. IEEE, 2003.
- 15 Julia Kempe and Thomas Vidick. Parallel repetition of entangled games. In *Proceedings of the 43rd Annual ACM Symposium on Theory of Computing*, pages 353–362. ACM, 2011.
- 16 Robert König, Renato Renner, and Christian Schaffner. The operational meaning of min- and max-entropy. *IEEE Transactions on Information Theory*, 55(9):4337–4347, 2009.
- 17 Ashwin Nayak and Julia Salzman. Limits on the ability of quantum states to convey classical messages. *Journal of the ACM (JACM)*, 53(1):184–206, 2006.
- 18 Michael A. Nielsen and Isaac L. Chuang. *Quantum computation and quantum information*. Cambridge university press, 2010.
- 19 Ran Raz. A parallel repetition theorem. *SIAM Journal on Computing*, 27(3):763–803, 1998.
- 20 Ran Raz. A counterexample to strong parallel repetition. *SIAM Journal on Computing*, 40(3):771–777, 2011.
- 21 Ricky Rosen. A  $k$ -provers parallel repetition theorem for a version of no-signaling model. *Discrete Math., Alg. and Appl.*, 2(4):457–468, 2010.
- 22 Mark M. Wilde. *Quantum information theory*. Cambridge University Press, 2013.
- 23 Dong Yang, Karol Horodecki, Michal Horodecki, Pawel Horodecki, Jonathan Oppenheim, and Wei Song. Squashed entanglement for multipartite states and entanglement measures based on the mixed convex roof. *IEEE Transactions on Information Theory*, 55(7):3375–3387, 2009.

# Upper Bounds on Quantum Query Complexity Inspired by the Elitzur-Vaidman Bomb Tester

Cedric Yen-Yu Lin and Han-Hsuan Lin

Center for Theoretical Physics, Massachusetts Institute of Technology  
77 Massachusetts Ave, Cambridge, MA 02139, USA  
{cedricl,hanmas}@mit.edu

---

## Abstract

Inspired by the Elitzur-Vaidman bomb testing problem [19], we introduce a new query complexity model, which we call bomb query complexity  $B(f)$ . We investigate its relationship with the usual quantum query complexity  $Q(f)$ , and show that  $B(f) = \Theta(Q(f)^2)$ .

This result gives a new method to upper bound the quantum query complexity: we give a method of finding bomb query algorithms from classical algorithms, which then provide nonconstructive upper bounds on  $Q(f) = \Theta(\sqrt{B(f)})$ . We subsequently were able to give explicit quantum algorithms matching our upper bound method. We apply this method on the single-source shortest paths problem on unweighted graphs, obtaining an algorithm with  $O(n^{1.5})$  quantum query complexity, improving the best known algorithm of  $O(n^{1.5}\sqrt{\log n})$  [21]. Applying this method to the maximum bipartite matching problem gives an  $O(n^{1.75})$  algorithm, improving the best known trivial  $O(n^2)$  upper bound.

**1998 ACM Subject Classification** F.2 Analysis of Algorithms and Problem Complexity

**Keywords and phrases** Quantum Algorithms, Query Complexity, Elitzur-Vaidman Bomb Tester, Adversary Method, Maximum Bipartite Matching

**Digital Object Identifier** 10.4230/LIPIcs.CCC.2015.537

## 1 Introduction

Quantum query complexity is an important method of understanding the power of quantum computers. In this model we are given a black-box containing a boolean string  $x = x_1 \cdots x_N$ , and we would like to calculate some function  $f(x)$  with as few quantum accesses to the black-box as possible. It is often easier to give bounds on the query complexity than to the time complexity of a problem, and insights from the former often prove useful in understanding the power and limitations of quantum computers. One famous example is Grover's algorithm for unstructured search [22]; by casting this problem into the query model it was shown that  $\Theta(\sqrt{N})$  queries was required [7], proving that Grover's algorithm is optimal.

Several methods have been proposed to bound the quantum query complexity. Upper bounds are almost always proven by finding better query algorithms. Some general methods of constructing quantum algorithms have been proposed, such as quantum walks [3, 45, 34, 28] and learning graphs [6]. For lower bounds, the main methods are the polynomial method [5] and adversary method [2]. In particular, the general adversary lower bound [27] has been shown to tightly characterize quantum query complexity [42, 43, 33], but calculating such a tight bound seems difficult in general. Nevertheless, the general adversary lower bound is valuable as a theoretical tool, for example in proving composition theorems [43, 33, 30] or showing nonconstructive (!) upper bounds [30].



© Cedric Yen-Yu Lin and Han-Hsuan Lin;  
licensed under Creative Commons License CC-BY  
30th Conference on Computational Complexity (CCC'15).

Editor: David Zuckerman; pp. 537–566



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

## Our work

To improve our understanding of quantum query complexity, we introduce and study an alternative oracle model, which we call the *bomb oracle* (see Section 3 for the precise definition). Our model is inspired by the concept of *interaction free measurements*, illustrated vividly by the Elitzur-Vaidman bomb testing problem [19], in which a property of a system can be measured without disturbing the system significantly. Like the quantum oracle model, in the bomb oracle model we want to evaluate a function  $f(x)$  on a hidden boolean string  $x = x_1 \cdots x_N$  while querying the oracle as few times as possible. In this model, however, the bomb oracle is a controlled quantum oracle with the extra requirement that the algorithm fails if the controlled query returns a 1. This seemingly impossible task can be tackled using the quantum Zeno effect [36], in a fashion similar to the Elitzur-Vaidman bomb tester [32] (Section 2.1).

Our main result (Theorem 4.1) is that the bomb query complexity,  $B(f)$ , is characterized by the square of the quantum query complexity  $Q(f)$ :

► **Theorem 4.1.**

$$B(f) = \Theta(Q(f)^2). \quad (1)$$

We prove the upper bound,  $B(f) = O(Q(f)^2)$  (Theorem 4.2), by adapting Kwiat et al.'s solution of the Elitzur-Vaidman bomb testing problem (Section 2.1, [32]) to our model. We prove the lower bound,  $B(f) = \Omega(Q(f)^2)$  (Theorem 4.3), by demonstrating that  $B(f)$  is lower bounded by the square of the general adversary bound [27],  $(\text{Adv}^\pm(f))^2$ . The aforementioned result that the general adversary bound tightly characterizes the quantum query complexity [42, 43, 33],  $Q(f) = \Theta(\text{Adv}^\pm(f))$ , allows us to draw our conclusion.

This characterization of Theorem 4.1 allows us to give *nonconstructive* upper bounds to the quantum query complexity for some problems. For some functions  $f$  a bomb query algorithm is easily designed by adapting a classical algorithm: specifically, we show that (stated informally):

► **Theorem 5.1 (informal).** Suppose there is a classical algorithm that computes  $f(x)$  in  $T$  queries, and the algorithm guesses the result of each query (0 or 1), making no more than an expected  $G$  mistakes for all  $x$ . Then we can design a bomb query algorithm that uses  $O(TG)$  queries, and hence  $B(f) = O(TG)$ . By our characterization of Theorem 4.1,  $Q(f) = O(\sqrt{TG})$ .

This result inspired us to look for an explicit quantum algorithm that reproduces the query complexity  $O(\sqrt{TG})$ . We were able to do so:

► **Theorem 5.2.** Under the assumptions of Theorem 5.1, there is an explicit algorithm (Algorithm F.1) for  $f$  with query complexity  $O(\sqrt{TG})$ .

Using Algorithm F.1, we were able to give an  $O(n^{3/2})$  algorithm for the single-source shortest paths (SSSP) problem in an unweighted graph with  $n$  vertices, beating the best-known  $O(n^{3/2}\sqrt{\log n})$  algorithm [21]. A more striking application is our  $O(n^{7/4})$  algorithm for maximum bipartite matching; in this case the best-known upper bound was the trivial  $O(n^2)$ , although the time complexity of this problem had been studied in [4] (and similar problems in [16]).

Finally, in Section 7 we briefly discuss a related query complexity model, which we call the *projective query complexity*  $P(f)$ , in which each quantum query to  $x$  is immediately followed by a classical measurement of the query result. This model seems interesting to us because

its power lies between classical and quantum: we observe that  $P(f) \leq B(f) = \Theta(Q(f)^2)$  and  $Q(f) \leq P(f) \leq R(f)$ , where  $R(f)$  is the classical randomized query complexity. We note that Regev and Schiff [41] showed that  $P(OR) = \Theta(N)$ .

## Past and related work

Mitchison and Jozsa have proposed a different computational model called *counterfactual computation* [37], also based on interaction-free measurement. In counterfactual computation the result of a computation may be learnt without ever running the computer. There has been some discussion on what constitutes counterfactual computation; see for example [26, 38, 25, 46, 24, 44, 47].

There have also been other applications of interaction-free measurement to quantum cryptography. For example, Noh has proposed counterfactual quantum cryptography [40], where a secret key is distributed between parties, even though a particle carrying secret information is not actually transmitted. More recently, Brodutch et al. proposed an adaptive attack [11] on Wiesner's quantum money scheme [48]; this attack is directly based off Kwiat et al.'s solution of the Elitzur-Vaidman bomb testing problem [32].

Our Algorithm F.1 is very similar to Kothari's algorithm for the oracle identification problem [31], and we refer to his analysis of the query complexity in our work.

The projective query model we detail in Section 7 was, to our knowledge, first considered by Aaronson in unpublished work in 2002 [1].

## Discussion and outlook

Our work raises a number of open questions. The most obvious ones are those pertaining to the application of our recipe for turning classical algorithms into bomb algorithms, Theorem 5.1:

- Can we generalize our method to handle non-boolean input and output? If so, we might be able to find better upper bounds for the adjacency-list model, or to study graph problems with weighted edges.
- Can our explicit (through Theorem 5.2) algorithm for maximum bipartite matching be made more *time* efficient? The best known quantum algorithm for this task has time complexity  $O(n^2 \log n)$  in the adjacency matrix model [4].
- Finally, can we find more upper bounds using Theorem 5.1? For example, could the query complexity of the maximum matching problem on general nonbipartite graphs be improved with Theorem 5.1, by analyzing the classical algorithm of Micali and Vazirani [35]?

Perhaps more fundamental, however, is the possibility that the bomb query complexity model will help us understand the relationship between the classical randomized query complexity,  $R(f)$ , and the quantum query complexity  $Q(f)$ . It is known [5] that for all total functions  $f$ ,  $R(f) = O(Q(f)^6)$ ; however, there is a long-standing conjecture that actually  $R(f) = O(Q(f)^2)$ . In light of our results, this conjecture is equivalent to the conjecture that  $R(f) = O(B(f))$ . Some more open questions, then, are the following:

- Can we say something about the relationship between  $R(f)$  and  $B(f)$  for specific classes of functions? For example, is  $R(f) = O(B(f)^2)$  for total functions?
- Referring to the notation of Theorem 5.1, we have  $B(f) = O(TG)$ . Is the quantity  $G$  related to other measures used in the study of classical decision-tree complexity, for example the certificate complexity, sensitivity [14], block sensitivity [39], or (exact or approximate) polynomial degree? (For a review, see [12].)

- What about other query complexity models that might help us understand the relationship between  $R(f)$  and  $Q(f)$ ? One possibility is the projective query complexity,  $P(f)$ , considered in Section 7. Regev and Schiff [41] have shown (as a special case of their results) that even with such an oracle,  $P(OR) = \Theta(N)$  queries are needed to evaluate the OR function.

We hope that further study on the relationship between bomb and classical randomized complexity will shed light on the power and limitations of quantum computation.

## 2 Preliminaries

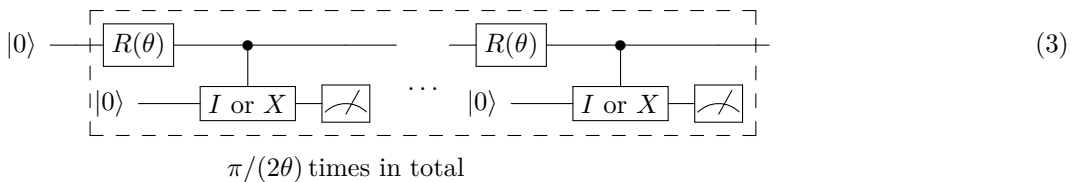
### 2.1 The Elitzur-Vaidman bomb testing problem

The Elitzur-Vaidman bomb testing problem [19] is a well-known thought experiment to demonstrate the possibility of *interaction free measurements*, a measurement on a property of a system without disturbing the system.

The bomb-testing problem is as follows: assume we have a bomb that is either a dud or a live bomb. The only way to interact with the bomb is to probe it with a photon: if the bomb is a dud, then the photon passes through unimpeded; if the bomb is live, then the bomb explodes. We would like to determine whether the bomb is live or not without exploding it. If we pass the photon through a beamsplitter before probing the bomb, we can implement the *controlled probe*, pictured below:



The controlled gate is  $I$  if the bomb is a dud, and  $X$  if it is live. [32] shows how to determine whether a bomb was live with arbitrarily low probability of explosion with the following scheme: writing  $R(\theta) = \exp(i\theta X)$ , the following circuit determines whether the bomb is live with failure probability  $O(\theta)$ :



If the bomb is a dud, then the controlled probes do nothing, and repeated application of  $R(\theta)$  rotates the control bit from  $|0\rangle$  to  $|1\rangle$ . If the bomb is live, the bomb explodes with  $O(\theta^2)$  probability in each application of the probe, projecting the control bit back to  $|0\rangle$ . After  $O(1/\theta)$  iterations the control bit stays in  $|0\rangle$ , with only a  $O(\theta)$  probability of explosion. Using  $O(1/\theta)$  operations, we can thus tell a dud bomb apart from a live one with only  $O(\theta)$  probability of explosion.

### 2.2 Quantum query complexity

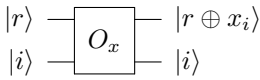
Throughout this paper, all functions  $f$  which we would like to calculate are assumed to have boolean input, i.e. the domain is  $D \subseteq \{0, 1\}^N$ .

For a boolean string  $x \in \{0, 1\}^N$ , the quantum oracle  $O_x$  is a unitary operator that acts on a one-qubit record register and an  $N$ -dimensional index register as follows ( $\oplus$  is the XOR



function):

$$O_x|r, i\rangle = |r \oplus x_i, i\rangle \tag{4}$$



The quantum query complexity  $Q_\delta(f)$  is the minimum number of applications of  $O_x$ 's in the circuit required to determine  $f(x)$  with error no more than  $\delta$  for all  $x$ . Since  $\delta$  can be amplified by majority voting, the choice of  $\delta$  only affects the query complexity by a  $\log(1/\delta)$  factor. We therefore often set  $\delta = 0.01$  and write  $Q_{0.01}(f)$  as  $Q(f)$ .

### 3 Bomb query complexity

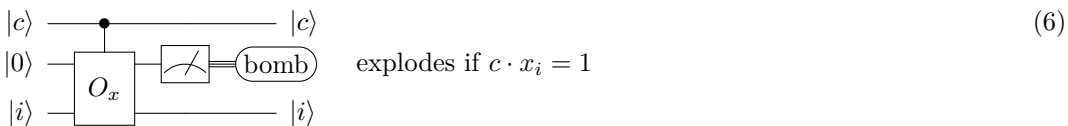
In this section we introduce a new query complexity model, which we call the *bomb query complexity*. A circuit in the bomb query model is a restricted quantum query circuit, with the following restrictions on the usage of the quantum oracle:

1. We have an extra control register  $|c\rangle$  used to control whether  $O_x$  is applied (we call the controlled version  $CO_x$ ):

$$CO_x|c, r, i\rangle = |c, r \oplus (c \cdot x_i), i\rangle. \tag{5}$$

where  $\cdot$  indicates boolean AND.

2. The record register,  $|r\rangle$  in the definition of  $CO_x$  above, *must* contain  $|0\rangle$  before  $CO_x$  is applied.
3. After  $CO_x$  is applied, the record register is immediately measured in the computational basis (giving the answer  $c \cdot x_i$ ), and the algorithm *terminates immediately if a 1 is measured* (if  $c \cdot x_i = 1$ ). We refer to this as *the bomb blowing up or the bomb exploding*.



We define the *bomb query complexity*  $B_{\epsilon, \delta}(f)$  to be the minimum number of times the above circuit needs to be applied in an algorithm such that the following hold for all input  $x$ :

- The algorithm reaches the end without the bomb exploding with probability at least  $1 - \epsilon$ . We refer to the probability that the bomb explodes as the *probability of explosion*.
- The total probability that the bomb either explodes or fails to output  $f(x)$  correctly is no more than  $\delta \geq \epsilon$ .

The above implies that the algorithm outputs the correct answer with probability at least  $1 - \delta$ .

We often set  $\delta = 0.01$ , and write simply  $B_\epsilon(f) = B_{\epsilon, 0.01}(f)$ . Sometimes we will even omit the  $\epsilon$ .

We will continue our discussion of the bomb query complexity in Appendix A. Note also that the definition of the bomb query complexity is inherently asymmetric with respect to 0 and 1 in the input, since the bomb explodes only on a 1. We will define a symmetric variant in Appendix A.2, although the proof that this variant is equivalent requires our main result, Theorem 4.1.

## 4 Main result

Our main result is the following:

► **Theorem 4.1.** *For all functions  $f$  with boolean input alphabet, and numbers  $\epsilon$  satisfying  $0 < \epsilon \leq 0.01$ ,*

$$B_{\epsilon,0.01}(f) = \Theta\left(\frac{Q_{0.01}(f)^2}{\epsilon}\right). \quad (7)$$

Here 0.01 can be replaced by any constant no more than 1/10.

**Proof.** The upper bound  $B_{\epsilon,\delta}(f) = O(Q_\delta(f)^2/\epsilon)$  is proved in Theorem 4.2. The lower bound  $B_{\epsilon,\delta}(f) = \Omega(Q_{0.01}(f)^2/\epsilon)$  is proved in Theorem 4.3. ◀

### 4.1 Upper bound

► **Theorem 4.2.** *For all functions  $f$  with boolean input alphabet, and numbers  $\epsilon, \delta$  satisfying  $0 < \epsilon \leq \delta \leq 1/10$ ,*

$$B_{\epsilon,\delta}(f) = O(Q_\delta(f)^2/\epsilon). \quad (8)$$

The proof follows the solution of Elitzur-Vaidman bomb-testing problem ([32], or Section 2.1). By taking advantage of the Quantum Zeno effect [36], using  $O(\frac{Q(f)}{\epsilon})$  calls to  $M_x$ , we can simulate one call to  $O_x$  with probability of explosion  $O(\frac{\epsilon}{Q(f)})$ . Replacing all  $O_x$  queries with this construction results in a bounded error algorithm with probability of explosion  $O(\frac{\epsilon}{Q(f)}Q(f)) = O(\epsilon)$ .

The complete proof is given in Appendix B.

### 4.2 Lower bound

► **Theorem 4.3.** *For all functions  $f$  with boolean input alphabet, and numbers  $\epsilon, \delta$  satisfying  $0 < \epsilon \leq \delta \leq 1/10$ ,*

$$B_{\epsilon,\delta}(f) = \Omega(Q_{0.01}(f)^2/\epsilon). \quad (9)$$

The proof of this result uses the generalized adversary bound  $\text{Adv}^\pm(f)$  [27]: we show that  $B_\epsilon(f) = \Omega(\text{Adv}^\pm(f)^2/\epsilon)$ , and then use the known result that  $Q(f) = O(\text{Adv}^\pm(f))$  [33]. The complete proof is given in Appendix C.

## 5 A general quantum algorithm inspired by $B(f)$

### 5.1 Using classical algorithms to design bomb query algorithms

We show *nonconstructive* upper bounds on  $Q(f)$  for some functions  $f$ , by creating bomb query algorithms and using that  $Q(f) = \Theta(\sqrt{\epsilon B_\epsilon(f)})$ , as the following theorem:

► **Theorem 5.1.** *Let  $f : D \rightarrow E$ , where  $D \subseteq \{0,1\}^N$ . Suppose there is a classical randomized query algorithm  $\mathcal{A}$ , that makes at most  $T$  queries, and evaluates  $f$  with bounded error. Let the query results of  $\mathcal{A}$  on random seed  $s_{\mathcal{A}}$  be  $x_{p_1}, x_{p_2}, \dots, x_{p_{\hat{T}(x)}}$ ,  $\hat{T}(x) \leq T$ , where  $x$  is the hidden query string.*

*Suppose there is another (not necessarily time-efficient) randomized algorithm  $\mathcal{G}$ , with random seed  $s_{\mathcal{G}}$ , which takes as input  $x_{p_1}, \dots, x_{p_{t-1}}$  and  $s_{\mathcal{A}}$ , and outputs a guess for the next*

query result  $x_{p_t}$  of  $\mathcal{A}$ . Assume that  $\mathcal{G}$  makes no more than an expected total of  $G$  mistakes (for all inputs  $x$ ). In other words,

$$\mathbf{E}_{s_{\mathcal{A}}, s_{\mathcal{G}}} \left\{ \sum_{t=1}^{\tilde{T}(x)} |\mathcal{G}(x_{p_1}, \dots, x_{p_{t-1}}, s_{\mathcal{A}}, s_{\mathcal{G}}) - x_{p_t}| \right\} \leq G \quad \forall x. \quad (10)$$

Note that  $\mathcal{G}$  is given the random seed  $s_{\mathcal{A}}$  of  $\mathcal{A}$ , so it can predict the next query index of  $\mathcal{A}$ . Then  $B_{\epsilon}(f) = O(TG/\epsilon)$ , and thus (by Theorem 4.1)  $Q(f) = O(\sqrt{TG})$ .

As an example, take  $f$  to be the OR function. One can easily find a classical algorithm with  $T = N$  (the algorithm takes at most  $N$  queries) and  $G = 1$  (the guessing algorithm always guesses the next query to be 0; since the algorithm terminates on a 1, it makes at most one mistake).

The proof idea is as follows: we take the classical algorithm and replace each classical query by the construction of Theorem 4.2 (see Eq. 29), using  $O(G/\epsilon)$  bomb queries each time. On each query, the bomb has a  $O(\epsilon/G)$  chance of exploding when the guess is wrong, and no chance of exploding when the guess is correct. Therefore the total probability of explosion is  $O(\epsilon/G) \cdot G = O(\epsilon)$ . The total number of bomb queries used is  $O(TG/\epsilon)$ .

For the full technical proof, see Appendix D.

## 5.2 Explicit quantum algorithm for Theorem 5.1

In this section we give an explicit quantum algorithm, in the setting of Theorem 5.1, that reproduces the given query complexity. This algorithm is very similar to the one given by R. Kothari for the oracle identification problem [31].

► **Theorem 5.2.** *Under the assumptions of Theorem 5.1, there is an explicit quantum algorithm for  $f$  with query complexity  $O(\sqrt{TG})$ .*

**Proof.** The explicit algorithm (Algorithm F.1) is given in Appendix F; we will give a high-level description shortly. We need the following quantum search algorithm as a subroutine:

► **Theorem 5.3** (Finding the first marked element in a list). *Suppose there is an ordered list of  $N$  elements, and each element is either marked or unmarked. Then there is a bounded-error quantum algorithm for finding the **first** marked element in the list (or determines that no marked elements exist), such that:*

- *If the first marked element is the  $d$ -th element of the list, then the algorithm uses an expected  $O(\sqrt{d})$  time and queries.*
- *If there are no marked elements, then the algorithm uses  $O(\sqrt{N})$  time and queries, but always determines correctly that no marked elements exist.*

This algorithm is straightforward to derive given the result in [18], and was already used in Kothari's algorithm [31]. We give the algorithm (Algorithm E.2) and its analysis in Appendix E.

We now describe our explicit quantum algorithm (Algorithm F.1 in Appendix F). The main idea for the algorithm is this: we first assume that the guesses made by  $\mathcal{G}$  are correct. By repeatedly feeding the output of  $\mathcal{G}$  back into  $\mathcal{A}$  and  $\mathcal{G}$ , we can obtain a list of query values for  $\mathcal{A}$  without any queries to the actual black box. We then search for the first deviation of the string  $x$  from the predictions of  $\mathcal{G}$ ; assuming the first deviation is the  $d_1$ -th query, by Theorem 5.3 the search takes  $O(\sqrt{d_1})$  queries (ignoring error for now). We then know that

all the guesses made by  $\mathcal{G}$  are correct up to the  $(d_1 - 1)$ -th query, and incorrect for the  $d_1$ -th query.

With the corrected result of the first  $d_1$  queries, we now continue by assuming again the guesses made by  $\mathcal{G}$  are correct starting from the  $(d_1 + 1)$ -th query, and search for the location of the next deviation,  $d_2$ . This takes  $O(\sqrt{d_2 - d_1})$  queries; we then know that all the guesses made by  $\mathcal{G}$  are correct from the  $(d_1 + 1)$ -th to  $(d_2 - 1)$ -th query, and incorrect for the  $d_2$ -th one. Continuing in this manner, we eventually determine all query results of  $\mathcal{A}$  after an expected  $G$  iterations. The expected number of queries is

$$O\left(\sum_{i=1}^G \sqrt{d_i - d_{i-1}}\right) = O(\sqrt{TG}) \quad (11)$$

by the Cauchy-Schwarz inequality.<sup>1</sup> ◀

Note that while Algorithm F.1 has query complexity  $O(\sqrt{TG})$ , the time complexity may be much higher. After all, Algorithm F.1 proceeds by simulating  $\mathcal{A}$  query-by-query, although the number of actual queries to the oracle is smaller. Whether or not we can get a algorithm faster than  $\mathcal{A}$  using this approach may depend on the problem at hand.

## 6 Improved upper bounds on quantum query complexity

We now use Theorem 5.2 to improve the quantum query complexity of certain graph problems.

### 6.1 Single source shortest paths for unweighted graphs

► **Problem 6.1** (Single source shortest paths (SSSP) for unweighted graphs). The adjacency matrix of a directed graph  $n$ -vertex graph  $G$  is provided as a black box. Given a fixed vertex  $v_{start}$ , our task is to find the lengths of the shortest paths from  $v_{start}$  to all other vertices  $w$  in  $G$ .

► **Theorem 6.2.** *The quantum query complexity of single-source shortest paths in an unweighted graph is  $\Theta(n^{3/2})$  in the adjacency matrix model.*

**Proof.** The lower bound of  $\Omega(n^{3/2})$  is known [17]. We show the upper bound by applying Theorem 5.2 to the breadth-first search (BFS) algorithm. Although  $T = O(n^2)$  queries are required for BFS in the worst case, if we always guess that  $(v, w)$  is not an edge, then the algorithm only needs to make  $G = n - 1$  mistakes (find  $n - 1$  actual edges) to construct the BFS tree. Therefore  $Q(f) = O(\sqrt{TG}) = O(n^{3/2})$ . ◀

The previous best known quantum algorithm for unweighted SSSP, to our best knowledge, was given by Furrow [21]; that algorithm has query complexity  $O(n^{3/2}\sqrt{\log n})$ .

We now consider the quantum query complexity of unweighted  $k$ -source shortest paths (finding  $k$  shortest-path trees rooted from  $k$  beginning vertices). If we apply BFS on  $k$  different starting vertices, then the expected number of wrong guesses is no more than  $G = k(n - 1)$ ; however, the total number of edges we query need not exceed  $T = O(n^2)$ , since an edge never needs to be queried more than once. Therefore

<sup>1</sup> It may seem like we actually need an extra logarithmic factor in the query complexity to keep the total error constant. However, Kothari showed [31] that multiple calls to Algorithm E.2 can be composed without an extra logarithmic factor.

► **Corollary 6.3.** *The quantum query complexity of unweighted  $k$ -source shortest paths in the adjacency matrix model is  $O(k^{1/2}n^{3/2})$ , where  $n$  is the number of vertices.*

We use this idea – that  $T$  need not exceed  $O(n^2)$  when dealing with graph problems – again in the following section.

## 6.2 Maximum bipartite matching

► **Problem 6.4** (Maximum bipartite matching). We are given as black box the adjacency matrix of an  $n$ -vertex undirected bipartite graph  $G = (V = X \cup Y, E)$ . A *matching* of  $G$  is a list of edges of  $G$  that do not share vertices. Our task is to find a maximum matching of  $G$ , i.e. a matching that contains the largest possible number of edges.

► **Theorem 6.5.** *The quantum query complexity of maximum bipartite matching is  $O(n^{7/4})$  in the adjacency matrix model, where  $n$  is the number of vertices.*

The proof proceeds by analyzing the classical Hopcroft-Karp algorithm [23], which uses up to  $O(\sqrt{n})$  iterations of modified breadth-first search and depth-first search. It will therefore turn out that  $G = O(\sqrt{n} \cdot n) = O(n^{3/2})$ ; however,  $T = O(n^2)$ , since no edge needs to be queried more than once. This gives  $Q = O(\sqrt{TG}) = O(n^{7/4})$ .

We give the complete proof in Appendix G.

To our knowledge, this is the first known nontrivial upper bound on the query complexity of maximum bipartite matching.<sup>2</sup> The time complexity of this problem was studied by Ambainis and Spalek in [4]; they gave an upper bound of  $O(n^2 \log n)$  time in the adjacency matrix model. A lower bound of  $\Omega(n^{3/2})$  for the query complexity of this problem was given in [9, 49].

For readers familiar with network flow, the arguments in this section also apply to Dinic's algorithm for maximum flow [15] on graphs with unit capacity, i.e. where the capacity of each edge is 0 or 1. On graphs with unit capacity, Dinic's algorithm is essentially the same as Hopcroft-Karp's, except that augmenting paths are over a general, nonbipartite flow network. (The set  $S$  in Step 2(c) of Algorithm G.1 is generally referred to as a *blocking flow* in this context.) It can be shown that only  $O(\min\{m^{1/2}, n^{2/3}\})$  iterations of Step 2 are required [29, 20], where  $m$  is the number of edges of the graph. Thus  $T = O(n^2)$ ,  $G = O(\min\{m^{1/2}, n^{2/3}\}n)$ , and therefore

► **Theorem 6.6.** *The quantum query complexity of the maximum flow problem in graphs with unit capacity is  $O(\min\{n^{3/2}m^{1/4}, n^{11/6}\})$ , where  $n$  and  $m$  are the number of vertices and edges in the graph, respectively.*

It is an open question whether a similar result for maximum matching in a general nonbipartite graph can be proven, perhaps by applying Theorem 5.2 to the classical algorithm of Micali and Vazirani [35].

## 7 Projective query complexity

We end this paper with a brief discussion on another query complexity model, which we will call the *projective query complexity*. This model is similar to the bomb query model in that the only way of accessing  $x_i$  is through a classical measurement; however, in the

<sup>2</sup> The trivial upper bound is  $O(n^2)$ , where all pairs of vertices are queried.

projective query model the algorithm does not terminate if a 1 is measured. Our motivation for considering the projective query model is that its power is intermediate between the classical and quantum query models. To the best of our knowledge, this model was first considered in 2002 in unpublished results by S. Aaronson [1].

A circuit in the projective query complexity model is a restricted quantum query circuit, with the following restrictions on the use of the quantum oracle:

1. We have an extra control register  $|c\rangle$  used to control whether  $O_x$  is applied (we call the controlled version  $CO_x$ ):

$$CO_x|c, r, i\rangle = |c, r \oplus (c \cdot x_i), i\rangle. \tag{12}$$

where  $\cdot$  indicates boolean AND.

2. The record register,  $|r\rangle$  in the definition of  $CO_x$  above, *must* contain  $|0\rangle$  before  $CO_x$  is applied.
3. After  $CO_x$  is applied, the record register is immediately measured in the computational basis, giving the answer  $c \cdot x_i$ . The result, a classical bit, can then be used to control further quantum unitaries (although only controlling the next unitary is enough, since the classical bit can be stored).



We wish to evaluate a function  $f(x)$  with as few calls to this *projective oracle* as possible. Let the number of oracle calls required to evaluate  $f(x)$ , with at most  $\delta$  error, be  $P_\delta(f)$ . By gap amplification, the choice of  $\delta$  only affects  $P_\delta(f)$  by a factor of  $\log(1/\delta)$ , and thus we will often omit  $\delta$ .

We can compare the definition in this section with the definition of the bomb query complexity in Section 3: the only difference is that if  $c \cdot x_i = 1$ , the algorithm terminates in the bomb model, while the algorithm can continue in the projective model. Therefore the following is evident:

► **Observation 7.1.**  $P_\delta(f) \leq B_{\epsilon, \delta}(f)$ , and therefore  $P(f) = O(Q(f)^2)$ .

Moreover, it is clear that the projective query model has power intermediate between classical and quantum (a controlled query in the usual quantum query model can be simulated by appending a 0 to the input string), and therefore letting  $R_\delta(f)$  be the classical randomized query complexity,

► **Observation 7.2.**  $Q_\delta(f) \leq P_\delta(f) \leq R_\delta(f)$ .

For explicit bounds on  $P$ , Regev and Schiff [41] have shown that for computing the OR function, the projective query complexity loses the Grover speedup:

► **Theorem 7.3** ([41]).  $P(OR) = \Omega(N)$ .

Note that this result says nothing about  $P(AND)$ , since the definition of  $P(f)$  is asymmetric with respect to 0 and 1 in the input.<sup>3</sup>

<sup>3</sup> We could have defined a symmetric version of  $P$ , say  $\tilde{P}$ , by allowing an extra guess on the measurement result, similar to our construction of  $\tilde{B}$  in Section A.2. Unfortunately, Regev and Schiff's result, Theorem 7.3, do not apply to this case, and we see no obvious equivalence between  $P$  and  $\tilde{P}$ .

We observe that there could be a separation in both parts of the inequality  $Q \leq P \leq B$ :

$$\begin{aligned} Q(OR) &= \Theta(\sqrt{N}), & P(OR) &= \Theta(N), & B(OR) &= \Theta(N) \\ Q(PARITY) &= \Theta(N), & P(PARITY) &= \Theta(N), & B(PARITY) &= \Theta(N^2) \end{aligned}$$

In the latter equation we used the fact that  $Q(PARITY) = \Theta(N)$  [5]. It therefore seems difficult to adapt our lower bound method in Section 4.2 to  $P(f)$ .

It would be interesting to find a general lower bound for  $P(f)$ , or to establish more clearly the relationship between  $Q(f)$ ,  $P(f)$ , and  $R(f)$ .

**Acknowledgements.** We are grateful to Scott Aaronson and Aram Harrow for many useful discussions, and Scott Aaronson and Shelby Kimmel for valuable suggestions on a preliminary draft. We also thank Andrew Childs for giving us permission to make use of his online proof of the general adversary lower bound in [13]. Special thanks to Robin Kothari for pointing us to his paper [31], and in particular his analysis showing that logarithmic factors can be removed from the query complexity of Algorithm F.1. We also thank the anonymous referees from QIP and CCC for their helpful comments. This work is supported by the ARO grant Contract Number W911NF-12-0486. CYL gratefully acknowledges support from the Natural Sciences and Engineering Research Council of Canada.

---

## References

- 1 Scott Aaronson. Personal communication, 2014.
- 2 Andris Ambainis. Quantum lower bounds by quantum arguments. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing (STOC)*, pages 636–643, 2000.
- 3 Andris Ambainis. Quantum walk algorithm for element distinctness. *SIAM Journal on Computing*, 37(1):210–239, 2007.
- 4 Andris Ambainis and Robert Špalek. Quantum algorithms for matching and network flows. In *Lecture Notes in Computer Science*, volume 3884, pages 172–183. Springer, 2006.
- 5 Robert Beals, Harry Buhrman, Richard Cleve, Michele Mosca, and Ronald de Wolf. Quantum lower bounds by polynomials. In *Proceedings of the 39th Annual Symposium on Foundations of Computer Science (FOCS)*, page 352, 1998.
- 6 Aleksandrs Belovs. Span programs for functions with constant-sized 1-certificates. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC)*, pages 77–84, 2012.
- 7 Charles H. Bennett, Ethan Bernstein, Gilles Brassard, and Umesh Vazirani. Strengths and weaknesses of quantum computing. *SIAM Journal on Computing*, 26(5):1510–1523, 1997.
- 8 Claude Berge. Two theorems in graph theory. *Proceedings of the National Academy of Sciences of the United States of America*, 43(9):842–844, 1957.
- 9 Aija Berzina, Andrej Dubrovsky, Rusins Freivalds, Lelde Lace, and Oksana Scegulnaja. Quantum query complexity for some graph problems. In *Lecture Notes in Computer Science*, volume 2932, pages 140–150. Springer, 2004.
- 10 Rajendra Bhatia. *Matrix Analysis*. Springer-Verlag, 1997.
- 11 Aharon Brodutch, Daniel Nagaj, Or Sattath, and Dominique Unruh. An adaptive attack on Wiesner’s quantum money. *arXiv preprint arXiv:1404.1507 [quant-ph]*, 2014.
- 12 Harry Buhrman and Ronald De Wolf. Complexity measures and decision tree complexity: A survey. *Theoretical Computer Science*, 288:2002, 1999.
- 13 Andrew Childs. <http://www.math.uwaterloo.ca/~amchilds/teaching/w13/l15.pdf>, 2013.



- 14 Stephen Cook, Cynthia Dwork, and Rüdiger Reischuk. Upper and lower time bounds for parallel random access machines without simultaneous writes. *SIAM Journal on Computing*, 15(1):87–97, 1986.
- 15 E. A. Dinic. Algorithm for solution of a problem of maximum flow in a network with power estimation. *Soviet Math Doklady*, 11:1277–1280, 1970.
- 16 Sebastian Dörn. Quantum algorithms for matching problems. *Theory of Computing Systems*, 45(3):613–628, October 2009.
- 17 Christoph Dürr, Mark Heiligman, Peter Høyer, and Mehdi Mhalla. Quantum query complexity of some graph problems. arXiv:quant-ph/0401091, 2004.
- 18 Christoph Dürr and Peter Høyer. A quantum algorithm for finding the minimum. *arXiv preprint arXiv:quant-ph/9607014*, 1996.
- 19 Avshalom C. Elitzur and Lev Vaidman. Quantum mechanical interaction-free measurements. *Foundations of Physics*, 23(7):987–997, July 1993.
- 20 Shimon Even and R. Endre Tarjan. Network flow and testing graph connectivity. *SIAM Journal on Computing*, 4(4):507–518, 1975.
- 21 Bartholomew Furrow. A panoply of quantum algorithms. *Quantum Information and Computation*, 8(8):834–859, September 2008.
- 22 Lov K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the 28th Annual ACM Symposium on the Theory of Computing (STOC)*, May 1996.
- 23 John E. Hopcroft and Richard M. Karp. An  $n^{5/2}$  algorithm for maximum matchings in bipartite graphs. *SIAM Journal on Computing*, 2(4):225–231, 1973.
- 24 Onur Hosten and Paul G. Kwiat. Weak measurements and counterfactual computation. *arXiv preprint arXiv:quant-ph/0612159*, 2006.
- 25 Onur Hosten, Matthew T. Rakher, Julio T. Barreiro, Nicholas A. Peters, and Paul Kwiat. Counterfactual computation revisited. *arXiv preprint arXiv:quant-ph/0607101*, 2006.
- 26 Onur Hosten, Matthew T. Rakher, Julio T. Barreiro, Nicholas A. Peters, and Paul G. Kwiat. Counterfactual quantum computation through quantum interrogation. *Nature*, 439:949–952, February 2006.
- 27 Peter Høyer, Troy Lee, and Robert Špalek. Negative weights make adversaries stronger. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing (STOC)*, pages 526–535, 2007.
- 28 Stacey Jeffery, Robin Kothari, and Frederic Magniez. Nested quantum walks with quantum data structures. In *Proceedings of the 24th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1474–1485, 2012.
- 29 Alexander V. Karzanov. O nakhozhdennii maksimal'nogo potoka v setyakh spetsial'nogo vida i nekotorykh prilozheniyakh. In L.A. Lyusternik, editor, *Matematicheskie Voprosy Upravleniya Proizvodstvom*, volume 5, pages 81–94. Moscow State University Press, 1973.
- 30 Shelby Kimmel. Quantum adversary (upper) bound. *Chicago Journal of Theoretical Computer Science*, 2013(4), 2013.
- 31 Robin Kothari. An optimal quantum algorithm for the oracle identification problem. In Ernst W. Mayr and Natacha Portier, editors, *Proceedings of the 31st International Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 25 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 482–493, Dagstuhl, Germany, 2014. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.
- 32 Paul Kwiat, Harald Weinfurter, Thomas Herzog, Anton Zeilinger, and Mark A. Kasevich. Interaction-free measurement. *Physical Review Letters*, 74(24):4763, 1995.
- 33 Troy Lee, Rajat Mittal, Ben W. Reichardt, Robert Špalek, and Mario Szegedy. Quantum query complexity of state conversion. In *Proceedings of the 52nd IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 344–353, 2011.

- 34 Frédéric Magniez, Ashwin Nayak, Jérémie Roland, and Miklos Santha. Search via quantum walk. *SIAM Journal on Computing*, 40(1):142–164, 2011.
- 35 Silvio Micali and Vijay V. Vazirani. An  $O(\sqrt{|V|} \cdot |E|)$  algorithm for finding maximum matching in general graphs. In *Proceedings of the 21st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 17–27, 1980.
- 36 B. Misra and E. C. G. Sudarshan. The Zeno’s paradox in quantum theory. *Journal of Mathematical Physics*, 18(4):756, 1977.
- 37 Graeme Mitchison and Richard Jozsa. Counterfactual computation. *Proceedings of the Royal Society A*, 457(2009):1175–1194, 2001.
- 38 Graeme Mitchison and Richard Jozsa. The limits of counterfactual computation. *arXiv preprint arXiv:quant-ph/0606092*, 2006.
- 39 Noam Nisan. CREW PRAMs and decision trees. *SIAM Journal on Computing*, 20(6):999–1007, 1991.
- 40 Tae-Gon Noh. Counterfactual quantum cryptography. *Physical Review Letters*, 103:230501, 2009.
- 41 Oded Regev and Liron Schiff. Impossibility of a quantum speed-up with a faulty oracle. In *Lecture Notes in Computer Science*, volume 5125, pages 773–781. Springer, 2008.
- 42 Ben W. Reichardt. Span programs and quantum query complexity: The general adversary bound is nearly tight for every boolean function. In *Proceedings of the 50th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 544–551, 2009.
- 43 Ben W. Reichardt. Reflections for quantum query algorithms. In *Proceedings of the 22nd ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 560–569, 2011.
- 44 Hatim Salih, Zheng-Hong Li, M. Al-Amri, and M. Suhail Zubairy. Protocol for direct counterfactual quantum communication. *Physical Review Letters*, 110:170502, 2013.
- 45 Mario Szegedy. Quantum speed-up of Markov chain based algorithms. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2004.
- 46 Lev Vaidman. The impossibility of the counterfactual computation for all possible outcomes. *arXiv preprint arXiv:quant-ph/0610174*, 2006.
- 47 Lev Vaidman. Comment on "protocol for direct counterfactual quantum communication" [arxiv:1206.2042]. *arXiv preprint arXiv:1304.6689 [quant-ph]*, 2013.
- 48 Stephen Wiesner. Conjugate coding. *ACM SIGACT News*, 15(1), 1983.
- 49 Shengyu Zhang. On the power of Ambainis’s lower bounds. *Theoretical Computer Science*, 339(2-3):241–256, 2005.

## **A** A more detailed discussion of bomb query complexity

We will continue our discussion of bomb query complexity from Section 3, to provide further intuition and also alternative characterizations that will be useful for the proofs contained in this appendix.

### **A.1** Properties of the bomb query complexity

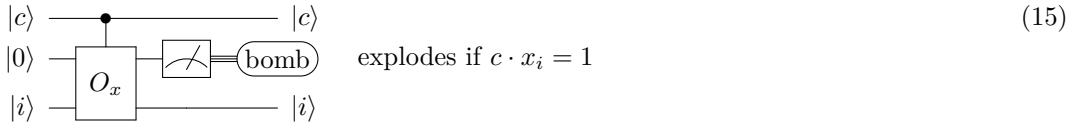
Recall that a circuit in the bomb query model has the following restrictions on the usage of the quantum oracle:

1. We have an extra control register  $|c\rangle$  used to control whether  $O_x$  is applied (we call the controlled version  $CO_x$ ):

$$CO_x|c, r, i\rangle = |c, r \oplus (c \cdot x_i), i\rangle. \quad (14)$$

where  $\cdot$  indicates boolean AND.

2. The record register,  $|r\rangle$  in the definition of  $CO_x$  above, *must* contain  $|0\rangle$  before  $CO_x$  is applied.
3. After  $CO_x$  is applied, the record register is immediately measured in the computational basis (giving the answer  $c \cdot x_i$ ), and the algorithm *terminates immediately if a 1 is measured* (if  $c \cdot x_i = 1$ ). We refer to this as *the bomb blowing up* or *the bomb exploding*.



We define the *bomb query complexity*  $B_{\epsilon, \delta}(f)$  to be the minimum number of times the above circuit needs to be applied in an algorithm such that the following hold for all input  $x$ :

- The algorithm reaches the end without the bomb exploding with probability at least  $1 - \epsilon$ . We refer to the probability that the bomb explodes as the *probability of explosion*.
- The total probability that the bomb either explodes or fails to output  $f(x)$  correctly is no more than  $\delta \geq \epsilon$ .

The above implies that the algorithm outputs the correct answer with probability at least  $1 - \delta$ .

The effect of the above circuit is equivalent to applying the following projector on  $|c, i\rangle$ :

$$M_x = CP_{x,0} = \sum_{i=1}^N |0, i\rangle\langle 0, i| + \sum_{x_i=0} |1, i\rangle\langle 1, i| \tag{16}$$

$$= I - \sum_{x_i=1} |1, i\rangle\langle 1, i|. \tag{17}$$

$CP_{x,0}$  (which we will just call  $M_x$  in our proofs later on) is the controlled version of  $P_{x,0}$ , the projector that projects onto the indices  $i$  on which  $x_i = 0$ :

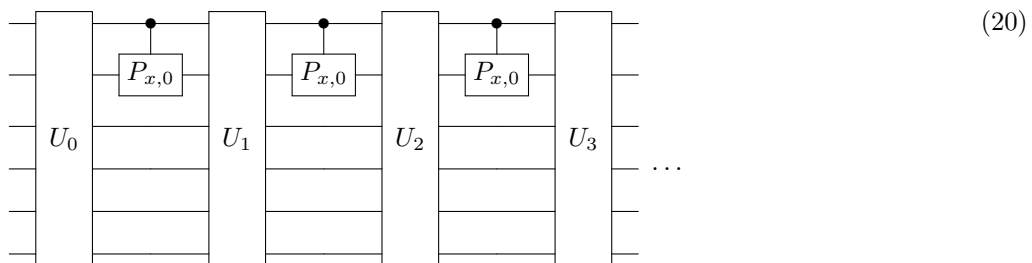
$$P_{x,0} = \sum_{x_i=0} |i\rangle\langle i|. \tag{18}$$

Thus Circuit 15 is equivalent to the following circuit :



In this notation, the square of the norm of a state is the probability that the state has survived to this stage, i.e. the algorithm has not terminated. The norm of  $(1 - c \cdot x_i)|x_i\rangle$  is 1 if  $c \cdot x_i = 0$  (the state survives this stage), and 0 otherwise (the bomb blows up).

A general circuit in this model looks like the following:



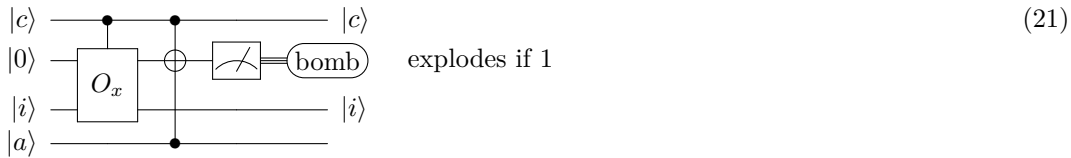
It is not at all clear that gap amplification can be done efficiently in the bomb query model to improve the error  $\delta$ ; after all, repeating the circuit multiple times increases the

chance that the bomb blows up. However, it turns out that the complexity  $B_{\epsilon,\delta}(f)$  is closely related to  $Q_\delta(f)$ , and therefore the choice of  $\delta$  affects  $B_{\epsilon,\delta}(f)$  by at most a  $\log^2(1/\delta)$  factor as long as  $\delta \geq \epsilon$  (this follows from the main result, Theorem 4.1). We therefore often omit  $\delta$  by setting  $\delta = 0.01$ , and write  $B_{\epsilon,0.01}(f)$  as  $B_\epsilon(f)$ . Sometimes we even omit the  $\epsilon$ .

### A.2 A symmetric variant of the bomb query complexity

Note that the definition of the bomb query complexity  $B(f)$  is inherently *asymmetric* with respect to 0 and 1 in the input: querying 1 causes the bomb to blow up, while querying 0 is safe. We will now define a *symmetric* bomb query model and its corresponding query complexity,  $\tilde{B}_{\epsilon,\delta}(f)$ . We will also show (using the main result, Theorem 4.1) that this definition is equivalent to the asymmetric version:  $\tilde{B}_{\epsilon,\delta}(f) = \Theta(B_{\epsilon,\delta}(f))$  for constant  $\delta$ .

We consider modifying the bomb query model as follows. We require that the input string  $x$  can only be accessed by the following circuit:



Compare with Circuit 15; the difference is that there is now an extra register  $|a\rangle$ , and the bomb explodes only if both  $x_i = a$  and the control bit is 1. In other words, the bomb explodes if  $c \cdot (x_i \oplus a) = 1$ . The three registers  $c$ ,  $i$ , and  $a$  are allowed to be entangled, however. If we discard the second register afterwards, the effect of this circuit, written as a projector, is

$$\tilde{M}_x = \sum_{i \in [N], a \in \{0,1\}} |0, i, a\rangle\langle 0, i, a| + \sum_{i, a: x_i = a} |1, i, a\rangle\langle 1, i, a|. \tag{22}$$

Let  $\tilde{B}_{\epsilon,\delta}(f)$  be the required number of queries to this modified bomb oracle  $\tilde{M}_x$  to calculate  $f(x)$  with error no more than  $\delta$ , with a probability of explosion no more than  $\epsilon$ . Using Theorem 4.1, we show that  $\tilde{B}$  and  $B$  are equivalent up to a constant:

► **Lemma A.1.** *If  $f : D \rightarrow E$ , where  $D \subseteq \{0,1\}^N$ , and  $\delta \leq 1/10$  is a constant, then  $B_{\epsilon,\delta}(f) = \Theta(\tilde{B}_{\epsilon,\delta}(f))$ .*

**Proof.** It should be immediately obvious that  $B_{\epsilon,\delta}(f) \geq \tilde{B}_{\epsilon,\delta}(f)$ , since a query in the  $B$  model can be simulated by a query in the  $\tilde{B}$  model by simply setting  $a = 0$ . In the following we show that  $B_{\epsilon,\delta}(f) = O(\tilde{B}_{\epsilon,\delta}(f))$ .

For each string  $x \in \{0,1\}^N$ , define the string  $\tilde{x} \in \{0,1\}^{2N}$  by concatenating two copies of  $x$  and flipping every bit of the second copy. In other words,

$$\tilde{x}_i = \begin{cases} x_i & \text{if } i \leq N \\ 1 - x_{i-N} & \text{if } i > N \end{cases}. \tag{23}$$

Let  $\tilde{D} = \{\tilde{x} : x \in D\}$ . Given a function  $f : D \rightarrow \{0,1\}$ , define  $\tilde{f} : \tilde{D} \rightarrow \{0,1\}$  by  $\tilde{f}(\tilde{x}) = f(x)$ .

We claim that a  $\tilde{B}$  query to  $x$  can be simulated by a  $B$  query to  $\tilde{x}$ . This can be seen by comparing  $\tilde{M}_x$ :

$$\tilde{M}_x = \sum_{i \in [N], a} |0, i, a\rangle\langle 0, i, a| + \sum_{i \in [N], a: x_i = a} |1, i, a\rangle\langle 1, i, a|. \tag{24}$$

and  $M_{\tilde{x}}$ :

$$M_{\tilde{x}} = \sum_{\tilde{i} \in [2N]} |0, \tilde{i}\rangle \langle 0, \tilde{i}| + \sum_{\tilde{i} \in [2N]: \tilde{x}_i=0} |1, \tilde{i}\rangle \langle 1, \tilde{i}|. \tag{25}$$

Recalling the definition of  $\tilde{x}$  in 23, we see that these two projectors are exactly equal if we encode  $\tilde{i}$  as  $(i, a)$ , where  $i \equiv \tilde{i} \pmod N$  and  $a = \lfloor i/N \rfloor$ .

Since  $\tilde{f}(\tilde{x}) = f(x)$ , we thus have  $\tilde{B}_{\epsilon, \delta}(f) = B_{\epsilon, \delta}(\tilde{f})$ . Our result then readily follows; it can easily be checked that  $Q(f) = Q(\tilde{f})$ , and therefore by Theorem 4.1,

$$\begin{aligned} \tilde{B}_{\epsilon, \delta}(f) &= B_{\epsilon, \delta}(\tilde{f}) = \Theta\left(\frac{Q(\tilde{f})^2}{\epsilon}\right) \\ &= \Theta\left(\frac{Q(f)^2}{\epsilon}\right) \end{aligned} \tag{26}$$

◀

There are some advantages to allowing the projector  $\tilde{M}_x$  instead of  $M_x$ . First of all, the inputs 0 and 1 in  $x$  are finally manifestly symmetric, unlike that in  $M_x$  (the bomb originally blew up if  $x_i = 1$ , but not if  $x_i = 0$ ). Moreover, we now allow the algorithm to *guess* an answer to the query (this answer may be entangled with the index register  $i$ ), and the bomb blows up only if the guess is wrong, controlled on  $c$ . This flexibility may allow more leeway in designing algorithms for the bomb query model, as we soon utilize.

**B Proof of the upper bound for  $B(f)$  (Theorem 4.2)**

We restate and prove Theorem 4.2:

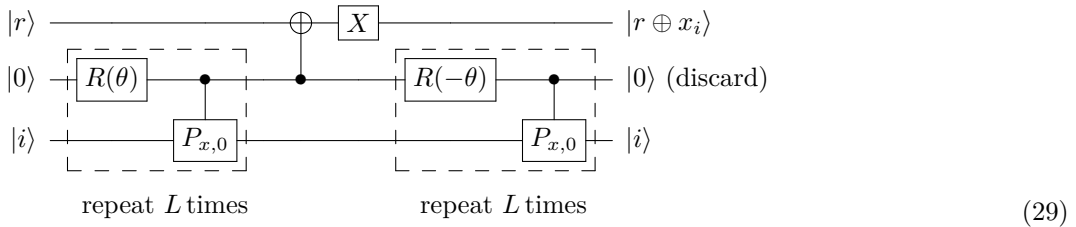
► **Theorem 4.2.** For all functions  $f$  with boolean input alphabet, and numbers  $\epsilon, \delta$  satisfying  $0 < \epsilon \leq \delta \leq 1/10$ ,

$$B_{\epsilon, \delta}(f) = O(Q_{\delta}(f)^2/\epsilon). \tag{27}$$

**Proof.** Let  $\theta = \pi/(2L)$  for some large positive integer  $L$  (chosen later), and let  $R(\theta)$  be the rotation

$$\begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix} \tag{28}$$

We claim that with  $2L$  calls to the bomb oracle  $M_x = CP_{x,0}$ , we can simulate  $O_x$  by the following circuit with probability of explosion less than  $\pi^2/(2L)$  and error  $O(1/L)$ .



In words, we simulate  $O_x$  acting on  $|r, i\rangle$  by the following steps:

1. Append an ancilla qubit  $|0\rangle$ , changing the state into  $|r, 0, i\rangle$ .
2. Repeat the following  $L$  times:

- a. apply  $R(\theta)$  on the second register
  - b. apply  $M_x$  on the third register controlled by the second register.
- At this point, if the bomb hasn't blown up, the second register should contain  $1 - x_i$ .
3. Apply  $CNOT$  on the first register controlled by the second register; this copies  $1 - x_i$  to the first register.
  4. Apply a  $NOT$  gate to the first register.
  5. Repeat the following  $L$  times to uncompute the second (ancilla) register :
    - a. apply  $R(-\theta)$  on the second register
    - b. apply  $M_x$  on the third register controlled by second register
  6. Discard the second (ancilla) register.

We now calculate explicitly the action of the circuit on an arbitrary state to confirm our claims above. Consider how the circuit acts on the basis state  $|r, 0, i\rangle$  (the second register being the appended ancilla). We break into cases:

- If  $x_i = 0$ , then  $P_{x,0}|i\rangle = |i\rangle$ , so the controlled projections do nothing. Thus in Step 2 the rotation  $R(\theta)^L = R(\pi/2)$  is applied to the ancilla qubit, rotating it from 0 to 1. After Step 2 then, the state is  $|r, 1, i\rangle$ . Step 3 and 4 together do not change the state, while Step 5 rotates the ancilla back to 0, resulting in the final state  $|r, 0, i\rangle$ .
- If  $x_i = 1$ , then  $P_{x,0}|i\rangle = 0$ , and

$$M_x|0, i\rangle = |0, i\rangle, \quad M_x|1, i\rangle = 0 \quad (\text{for } x_i = 1) \tag{30}$$

Therefore in Step 2 and Step 5, after each rotation  $R(\pm\theta)$ , the projection  $CP_{x,0}$  projects the ancilla back to 0:

$$M_x R(\theta)|0, i\rangle = M_x(\cos\theta|0\rangle + \sin\theta|1\rangle)|i\rangle = \cos\theta|0, i\rangle \quad (\text{for } x_i = 1) \tag{31}$$

Each application of  $M_x R(\theta)$  thus has no change on the state other than to shrink its amplitude by  $\cos\theta$ . The CNOT in Step 3 has no effect (since the ancilla stays in 0), and Step 4 maps  $|r\rangle$  to  $|r \oplus 1\rangle$ . Since there are  $2L$  applications of this shrinkage (in Step 2 and 5), the final state is  $\cos^{2L}\theta|r \oplus 1, 0, i\rangle$ .

We can now combine the two cases: by linearity, the application of the circuit on a general state  $\sum_{r,i} a_{r,i}|r, i\rangle$  (removing the ancilla) is

$$\sum_{r,i} a_{r,i}|r, i\rangle \rightarrow \sum_{r \in \{0,1\}, x_i=0} a_{r,i}|r, i\rangle + \sum_{r \in \{0,1\}, x_i=1} a_{r,i} \cos^{2L}(\theta)|r \oplus 1, i\rangle \tag{32}$$

$$= \sum_{r,i} a_{r,i} \cos^{2Lx_i} \left( \frac{\pi}{2L} \right) |r \oplus x_i, i\rangle \equiv |\psi'\rangle \tag{33}$$

Thus the effect of this construction simulates the usual quantum oracle  $|r, i\rangle \rightarrow |r \oplus x_i, i\rangle$  with probability of explosion no more than

$$1 - \cos^{4L} \left( \frac{\pi}{2L} \right) \leq 1 - \left( 1 - \frac{\pi^2}{4L^2} \right)^{2L} \leq \frac{\pi^2}{2L}. \tag{34}$$

Moreover, the difference between the output of our circuit,  $|\psi'\rangle$ , and the output on the quantum oracle,  $|\psi\rangle = \sum_{r,i} a_{r,i}|r \oplus x_i, i\rangle$ , is

$$\| |\psi'\rangle - |\psi\rangle \| = \left\| \sum_{r \in \{0,1\}, x_i=1} a_{r,i} (1 - \cos^{2L}(\theta)) |r \oplus 1, i\rangle \right\| \tag{35}$$

$$\leq 1 - \cos^{2L} \frac{\pi}{2L} \leq \frac{\pi^2}{4L}. \tag{36}$$

Given this construction, we can now prove our theorem. Suppose we are given a quantum algorithm that finds  $f(x)$  with  $Q_{\delta'}(f)$  queries, making at most  $\delta' = \delta - \epsilon$  error. We construct an algorithm using bomb oracles instead by replacing each of the applications of the quantum oracle  $O_x$  by our circuit construction (29), where we choose

$$L = \left\lceil \frac{\pi^2}{2\epsilon} Q_{\delta'}(f) \right\rceil \quad (37)$$

Then the probability of explosion is no more than

$$\frac{\pi^2}{2L} Q_{\delta'}(f) \leq \epsilon \quad (38)$$

and the difference between the final states,  $|\psi_f\rangle$  and  $|\psi'_f\rangle$ , is at most

$$\| |\psi'_f\rangle - |\psi_f\rangle \| \leq \frac{\pi^2}{4L} Q_{\delta'}(f) \leq \frac{\epsilon}{2}. \quad (39)$$

Therefore

$$\begin{aligned} |\langle \psi'_f | P | \psi'_f \rangle - \langle \psi_f | P | \psi_f \rangle| &\leq |\langle \psi'_f | P | \psi'_f \rangle - \langle \psi_f | P | \psi'_f \rangle| + |\langle \psi'_f | P | \psi_f \rangle - \langle \psi_f | P | \psi_f \rangle| \\ &\leq \| |\psi'_f\rangle \| \| P (|\psi'_f\rangle - |\psi_f\rangle) \| + \| P (|\psi'_f\rangle - |\psi_f\rangle) \| \| |\psi_f\rangle \| \\ &\leq \epsilon/2 + \epsilon/2 = \epsilon \end{aligned} \quad (40)$$

for any projector  $P$  (in particular, the projector that projects onto the classical answer at the end of the algorithm). The algorithm accumulates at most  $\epsilon$  extra error at the end, giving a total error of no more than  $\delta' + \epsilon = \delta$ . This algorithm makes  $2LQ_{\delta'}(f) < \frac{\pi^2}{\epsilon} Q_{\delta'}^2(f) + 2Q_{\delta'}(f)$  queries to the bomb oracle, and therefore

$$B_{\epsilon,\delta}(f) < \frac{\pi^2}{\epsilon} Q_{\delta-\epsilon}(f)^2 + 2Q_{\delta-\epsilon}(f) \quad (41)$$

$$= O\left(\frac{Q_{\delta-\epsilon}(f)^2}{\epsilon}\right). \quad (42)$$

From this we can derive that  $B_{\epsilon,\delta}(f) = O(Q_{\delta}(f)^2/\epsilon)$ :

$$\begin{aligned} B_{\epsilon,\delta}(f) &< B_{\epsilon/2,\delta}(f) \\ &= O\left(\frac{Q_{\delta-\epsilon/2}(f)^2}{\epsilon}\right), \quad \text{by 42} \\ &= O\left(\frac{Q_{\delta}(f)^2}{\epsilon}\right), \quad \text{since } \frac{\delta}{2} \leq \delta - \frac{\epsilon}{2}. \end{aligned} \quad (43)$$

◀

## C Proof of the adversary lower bound for $B(f)$ (Theorem 4.3)

Before we give the proof of the general result that  $B(f) = \Omega(Q(f)^2)$  (Theorem 4.3), we will illustrate the proof by means of an example, the special case where  $f$  is the AND function.

► **Theorem C.1.** For  $\delta < 1/10$ ,  $B_{\epsilon,\delta}(\text{AND}) = \Omega(\frac{N}{\epsilon})$ .



**Proof.** Let  $|\psi_t^0\rangle$  be the unnormalized state of the algorithm with  $x = 1^n$ , and  $|\psi_t^k\rangle$  be the unnormalized state with  $x = 1 \cdots 101 \cdots 1$ ,  $x_k = 0$ , right before the  $(t + 1)$ -th call to  $M_x$ . Then

$$|\psi_{t+1}^x\rangle = U_{t+1}M_x|\psi_t^x\rangle \tag{44}$$

for some unitary  $U_{t+1}$ . For ease of notation, we'll write  $M_0 \equiv M_{1^n}$  and  $M_k = M_{1 \cdots 101 \cdots 1}$ , where the  $k$ -th bit is 0 in the latter case. When acting on the control and index bits,

$$\begin{aligned} M_0 &= \sum_{i=1}^N |0, i\rangle\langle 0, i| \\ M_k &= \sum_{i=1}^N |0, i\rangle\langle 0, i| + |1, k\rangle\langle 1, k|. \end{aligned} \tag{45}$$

Since the  $M_i$ 's are projectors,  $M_i^2 = M_i$ . Define

$$\epsilon_t^i = \langle \psi_t^i | (I - M_i) | \psi_t^i \rangle, \quad i = 0, 1, \dots, N. \tag{46}$$

Note that  $\langle \psi_{t+1}^i | \psi_{t+1}^i \rangle = \langle \psi_t^i | M_i^2 | \psi_t^i \rangle = \langle \psi_t^i | M_i | \psi_t^i \rangle = \langle \psi_t^i | \psi_t^i \rangle - \epsilon_t^i$ , for all  $i = 0, \dots, N$  (including 0!), and hence

$$\sum_{t=0}^{T-1} \epsilon_t^i = \langle \psi_0^i | \psi_0^i \rangle - \langle \psi_T^i | \psi_T^i \rangle \leq \epsilon. \tag{47}$$

We now define the progress function. Let

$$W_t^k = \langle \psi_t^0 | \psi_t^k \rangle \tag{48}$$

and let the progress function be a sum over  $W^k$ 's:

$$W_t = \sum_{k=1}^N W_t^k = \sum_{k=1}^N \langle \psi_t^0 | \psi_t^k \rangle. \tag{49}$$

We can lower bound the total change in the progress function by (see [2] for a proof; their proof equally applies to unnormalized states)

$$W_0 - W_T \geq (1 - 2\sqrt{\delta(1 - \delta)})N. \tag{50}$$

We now proceed to upper bound  $W_0 - W_T$ . Note that

$$\begin{aligned} W_t^k - W_{t+1}^k &= \langle \psi_t^0 | \psi_t^k \rangle - \langle \psi_{t+1}^0 | \psi_{t+1}^k \rangle \\ &= \langle \psi_t^0 | (I - M_0)M_k | \psi_t^k \rangle + \langle \psi_t^0 | M_0(I - M_k) | \psi_t^k \rangle \\ &\quad + \langle \psi_t^0 | (I - M_0)(I - M_k) | \psi_t^k \rangle \end{aligned} \tag{51}$$

and since  $M_0(I - M_k) = 0$ ,  $(I - M_0)M_k = |1, k\rangle\langle 1, k|$ , we have

$$\begin{aligned} W_t^k - W_{t+1}^k &\leq \langle \psi_t^0 | 1, k \rangle \langle 1, k | \psi_t^k \rangle + \|(I - M_0) | \psi_t^0 \rangle\| \|(I - M_k) | \psi_t^k \rangle\| \\ &\leq \|\langle 1, k | \psi_t^0 \rangle\| + \sqrt{\epsilon_t^0 \epsilon_t^k}. \end{aligned} \tag{52}$$

where we used 46. Summing over  $k$  and  $t$ , we obtain

$$\begin{aligned}
 W_0 - W_T &\leq \sum_{t=0}^{T-1} \sum_{k=1}^N \left[ \|\langle 1, k | \psi_t^0 \rangle\| + \sqrt{\epsilon_t^0 \epsilon_t^k} \right] \\
 &\leq \sqrt{TN} \sqrt{\sum_{t=0}^{T-1} \sum_{k=1}^N \langle \psi_t^0 | 1, k \rangle \langle 1, k | \psi_t^0 \rangle} + \sum_{t=0}^{T-1} \sum_{k=1}^N \frac{\epsilon_t^0 + \epsilon_t^k}{2} \\
 &\leq \sqrt{TN} \sqrt{\sum_{t=0}^{T-1} \langle \psi_t^0 | (I - M_0) | \psi_t^0 \rangle} + N\epsilon \\
 &\leq \sqrt{TN \sum_{t=0}^{T-1} \epsilon_t^0} + N\epsilon \\
 &\leq \sqrt{\epsilon TN} + N\epsilon
 \end{aligned} \tag{53}$$

where in the second line we used Cauchy-Schwarz and the AM-GM inequality. Combined with  $W_0 - W_T \geq (1 - 2\sqrt{\delta(1-\delta)})N$  (Eq. 50), this immediately gives us

$$T \geq \frac{(1 - 2\sqrt{\delta(1-\delta)} - \epsilon)^2 N}{\epsilon}. \tag{54}$$

◀

We now proceed to prove the general result. This proof follows the presentation given in A. Childs’s online lecture notes [13], which we found quite illuminating.

► **Theorem 4.3.** For all functions  $f$  with boolean input alphabet, and numbers  $\epsilon, \delta$  satisfying  $0 < \epsilon \leq \delta \leq 1/10$ ,

$$B_{\epsilon, \delta}(f) = \Omega(Q_{0.01}(f)^2 / \epsilon). \tag{55}$$

**Proof.** We prove the lower bound on  $B_{\epsilon, \delta}$  by showing that it is lower bounded by  $\Omega(\text{Adv}^\pm(f)^2 / \epsilon)$ , where  $\text{Adv}^\pm(f)$  is the generalized (i.e. allowing negative weights) adversary bound [27] for  $f$ . We can then derive our theorem from the result [33] that  $Q(f) = O(\text{Adv}^\pm(f))$ .

We generalize the bound on the  $f = \text{AND}$  case to an adversary bound for  $B_{\epsilon, \delta}$  on arbitrary  $f$ . Define the projectors

$$\begin{aligned}
 \Pi_0 &= \sum_{i=1}^N |0, i\rangle\langle 0, i| \\
 \Pi_i &= |1, i\rangle\langle 1, i|, \quad i = 1, \dots, n.
 \end{aligned} \tag{56}$$

It is clear that

$$\Pi_0 + \sum_{i=1}^N \Pi_i = I. \tag{57}$$

Note that  $M_x = CP_{x,0}$  is

$$M_x = \Pi_0 + \sum_{i:x_i=0} \Pi_i. \tag{58}$$

Define  $|\psi_t^x\rangle$  as the state of the algorithm right before the  $(t+1)$ -th query with input  $x$ ; then

$$|\psi_{t+1}^x\rangle = U_{t+1}M_x|\psi_t^x\rangle \quad (59)$$

for some unitary  $U_{t+1}$ . Now if we let

$$\epsilon_t^x = \langle \psi_t^x | (I - M_x) | \psi_t^x \rangle \quad (60)$$

then it follows that  $\langle \psi_t^x | \psi_t^x \rangle - \langle \psi_{t+1}^x | \psi_{t+1}^x \rangle = \epsilon_t^x$ , and thus

$$\sum_{t=0}^{T-1} \epsilon_t^x = \langle \psi_0^x | \psi_0^x \rangle - \langle \psi_T^x | \psi_T^x \rangle \leq \epsilon. \quad (61)$$

We proceed to define the progress function. Let  $S$  be the set of allowable input strings  $x$ . Let  $\Gamma$  be an *adversary matrix*, i.e. an  $S \times S$  matrix such that

1.  $\Gamma_{xy} = \Gamma_{yx} \quad \forall x, y \in S$ ; and
2.  $\Gamma_{xy} = 0$  if  $f(x) \neq f(y)$ .

Let  $a$  be the normalized eigenvector of  $\Gamma$  with eigenvalue  $\pm \|\Gamma\|$ , where  $\pm \|\Gamma\|$  is the largest (by absolute value) eigenvalue of  $\Gamma$ . Define the progress function

$$W_t = \sum_{x, y \in S} \Gamma_{xy} a_x^* a_y \langle \psi_t^x | \psi_t^y \rangle. \quad (62)$$

For  $\epsilon \leq \delta < 1/10$  we have that<sup>4</sup> (see [27] for a proof; their proof applies equally well to unnormalized states)

$$|W_0 - W_T| \geq (1 - 2\sqrt{\delta(1-\delta)} - 2\delta)\|\Gamma\| \quad (63)$$

We now proceed to upper bound  $|W_0 - W_T| \leq \sum_t |W_t - W_{t-1}|$ . Note that

$$\begin{aligned} W_t - W_{t+1} &= \sum_{x, y \in S} \Gamma_{xy} a_x^* a_y (\langle \psi_t^x | \psi_t^y \rangle - \langle \psi_{t+1}^x | \psi_{t+1}^y \rangle) \\ &= \sum_{x, y \in S} \Gamma_{xy} a_x^* a_y (\langle \psi_t^x | \psi_t^y \rangle - \langle \psi_t^x | M_x M_y | \psi_t^y \rangle) \\ &= \sum_{x, y \in S} \Gamma_{xy} a_x^* a_y (\langle \psi_t^x | (I - M_x) M_y | \psi_t^y \rangle \\ &\quad + \langle \psi_t^x | M_x (I - M_y) | \psi_t^y \rangle + \langle \psi_t^x | (I - M_x)(I - M_y) | \psi_t^y \rangle) \end{aligned} \quad (64)$$

We bound the three terms separately. For the first two terms, use

$$\begin{aligned} (I - M_x)M_y &= \sum_{i: x_i=1, y_i=0} \Pi_i \\ &= (I - M_x) \sum_{i: x_i \neq y_i} \Pi_i \end{aligned} \quad (65)$$

Define the  $S \times S$  matrix  $\Gamma_i$  as

$$\Gamma_i = \begin{cases} \Gamma_{xy} & \text{if } x_i \neq y_i \\ 0 & \text{if } x_i = y_i \end{cases} \quad (66)$$

<sup>4</sup> As described in [27], the  $2\delta$  term can be removed if the output is boolean (0 or 1).

The first term of 64 is

$$\begin{aligned} \sum_{x,y \in S} \sum_{i: x_i \neq y_i} \Gamma_{xy} a_x^* a_y \langle \psi_t^x | (I - M_x) \Pi_i | \psi_t^y \rangle &= \sum_{x,y \in S} \sum_{i=1}^N (\Gamma_i)_{xy} a_x^* a_y \langle \psi_t^x | (I - M_x) \Pi_i | \psi_t^y \rangle \\ &= \sum_{i=1}^N \text{tr}(Q_i \Gamma_i \tilde{Q}_i^\dagger) \end{aligned} \quad (67)$$

where

$$Q_i = \sum_{x \in S} a_x \Pi_i | \psi_t^x \rangle \langle x | \quad (68)$$

$$\tilde{Q}_i = \sum_{x \in S} a_x \Pi_i (I - M_x) | \psi_t^x \rangle \langle x |. \quad (69)$$

Although both  $Q_i$  and  $\tilde{Q}_i$  depend on  $t$ , we suppress the  $t$  dependence in the notation. Similarly, the second term of 64 is equal to  $\sum_{i=1}^N \text{tr}(\tilde{Q}_i \Gamma_i Q_i^\dagger)$ . We can also rewrite the third term of 64 as

$$\sum_{x,y \in S} \Gamma_{xy} a_x^* a_y \langle \psi_t^x | (I - M_x) (I - M_y) | \psi_t^y \rangle = \text{tr}(Q' \Gamma Q'^\dagger) \quad (70)$$

where

$$Q' = \sum_{x \in S} a_x (I - M_x) | \psi_t^x \rangle \langle x |. \quad (71)$$

Therefore, adding absolute values,

$$|W_t - W_{t+1}| \leq \sum_{i=1}^N \left[ \left| \text{tr}(Q_i \Gamma_i \tilde{Q}_i^\dagger) \right| + \left| \text{tr}(\tilde{Q}_i \Gamma_i Q_i^\dagger) \right| \right] + \left| \text{tr}(Q' \Gamma Q'^\dagger) \right| \quad (72)$$

To continue, we need the following lemma:

► **Lemma C.2.** *For any  $m, n > 0$  and matrices  $X \in \mathbb{C}^{m \times n}$ ,  $Y \in \mathbb{C}^{n \times n}$ ,  $Z \in \mathbb{C}^{n \times m}$ , we have  $|\text{tr}(XYZ)| \leq \|X\|_F \|Y\| \|Z\|_F$ . Here  $\|\cdot\|$  and  $\|\cdot\|_F$  denote the spectral norm and Frobenius norm, respectively.*

This lemma can be proved by using that  $|\text{tr}(XYZ)| \leq \|Y\| \|ZX\|_{tr}$  and  $\|ZX\|_{tr} \leq \|X\|_F \|Z\|_F$ , which follows from [10, Exercise IV.2.12 and Corollary IV.2.6]. A more accessible proof is found online at [13].

Then by Lemma C.2,

$$\sum_{i=1}^N \left| \text{tr}(Q_i \Gamma_i \tilde{Q}_i^\dagger) \right| \leq \sum_{i=1}^N \|\Gamma_i\| \|Q_i\|_F \|\tilde{Q}_i\|_F \quad (73)$$

Since

$$\begin{aligned} \sum_{i=1}^N \|Q_i\|_F^2 &= \sum_{i=1}^N \sum_{x \in S} |a_x|^2 \|\Pi_i | \psi_t^x \rangle\|^2 \\ &= \sum_{x \in S} |a_x|^2 \langle \psi_t^x | \sum_{i=1}^N \Pi_i | \psi_t^x \rangle \\ &\leq \sum_{x \in S} |a_x|^2 \\ &= 1 \end{aligned} \quad (74)$$

and

$$\begin{aligned}
 \sum_{i=1}^N \|\tilde{Q}_i\|_F^2 &= \sum_{i=1}^N \sum_{x \in S} |a_x|^2 \|\Pi_i(I - M_x)|\psi_t^x\rangle\|^2 \\
 &= \sum_{x \in S} |a_x|^2 \langle \psi_t^x | (I - M_x) \left( \sum_{i=1}^N \Pi_i \right) (I - M_x) | \psi_t^x \rangle \\
 &\leq \sum_{x \in S} |a_x|^2 \langle \psi_t^x | (I - M_x) | \psi_t^x \rangle \\
 &= \sum_{x \in S} |a_x|^2 \epsilon_t^x
 \end{aligned} \tag{75}$$

we have, by Cauchy-Schwarz,

$$\sum_{i=1}^N \|Q_i\|_F \|\tilde{Q}_i\|_F \leq \sqrt{\sum_{x \in S} |a_x|^2 \epsilon_t^x} \tag{76}$$

Therefore by 73 and 76,

$$\sum_{i=1}^N \left| \text{tr}(Q_i \Gamma_i \tilde{Q}_i^\dagger) \right| \leq \sqrt{\sum_{x \in S} |a_x|^2 \epsilon_t^x} \max_{i \in [N]} \|\Gamma_i\|. \tag{77}$$

Similarly for  $\text{tr}(Q' \Gamma Q'^\dagger)$ , we have

$$\begin{aligned}
 \|Q'\|_F^2 &= \sum_{x \in S} |a_x|^2 \|(I - M_x)|\psi_t^x\rangle\|^2 \\
 &= \sum_{x \in S} |a_x|^2 \langle \psi_t^x | (I - M_x) | \psi_t^x \rangle \\
 &= \sum_{x \in S} |a_x|^2 \epsilon_t^x
 \end{aligned} \tag{78}$$

and using Lemma C.2,

$$\text{tr}(Q' \Gamma Q'^\dagger) \leq \|Q'\|_F^2 \|\Gamma\| \tag{79}$$

$$= \sum_{x \in S} |a_x|^2 \epsilon_t^x \|\Gamma\| \tag{80}$$

Thus continuing from 72, we have that

$$|W_t - W_{t+1}| \leq 2 \sqrt{\sum_{x \in S} |a_x|^2 \epsilon_t^x} \max_{i \in [N]} \|\Gamma_i\| + \sum_{x \in S} |a_x|^2 \epsilon_t^x \|\Gamma\| \tag{81}$$

Finally, if we sum the above over  $t$  we obtain

$$|W_0 - W_T| \leq 2 \max_{i \in [N]} \|\Gamma_i\| \sum_{t=0}^{T-1} \sqrt{\sum_{x \in S} |a_x|^2 \epsilon_t^x} + \sum_{t=0}^{T-1} \sum_{x \in S} |a_x|^2 \epsilon_t^x \|\Gamma\| \tag{82}$$

The first term can be bounded using Cauchy-Schwarz:

$$\begin{aligned}
 \sum_{t=0}^{T-1} \sqrt{\sum_{x \in S} |a_x|^2 \epsilon_t^x} &\leq \sqrt{T \sum_{t=0}^{T-1} \sum_{x \in S} |a_x|^2 \epsilon_t^x} \\
 &\leq \sqrt{\epsilon T}
 \end{aligned} \tag{83}$$

where we used  $\sum_t \epsilon_t^x \leq \epsilon$  and  $\sum_x |a_x|^2 = 1$ . The second term can be summed easily:

$$\begin{aligned} \sum_{t=0}^{T-1} \sum_{x \in S} |a_x|^2 \epsilon_t^x \|\Gamma\| &\leq \sum_{x \in S} |a_x|^2 \epsilon \|\Gamma\| \\ &= \epsilon \|\Gamma\|. \end{aligned} \quad (84)$$

Therefore

$$|W_0 - W_T| \leq 2\sqrt{\epsilon T} \max_{i \in [N]} \|\Gamma_i\| + \epsilon \|\Gamma\|. \quad (85)$$

Combined with our lower bound  $|W_0 - W_T| \geq (1 - 2\sqrt{\delta(1-\delta)} - 2\delta)\|\Gamma\|$ , this immediately gives

$$T \geq \frac{(1 - 2\sqrt{\delta(1-\delta)} - 2\delta - \epsilon)^2}{4\epsilon} \frac{\|\Gamma\|^2}{\max_{i \in [N]} \|\Gamma_i\|^2}. \quad (86)$$

Recalling that [27]

$$\text{Adv}^\pm(f) = \max_{\Gamma} \frac{\|\Gamma\|}{\max_{i \in [N]} \|\Gamma_i\|}, \quad (87)$$

we obtain<sup>5</sup>

$$T \geq \frac{(1 - 2\sqrt{\delta(1-\delta)} - 2\delta - \epsilon)^2}{4\epsilon} \text{Adv}^\pm(f)^2. \quad (88)$$

We now use the tight characterization of the quantum query complexity by the general weight adversary bound:

► **Theorem C.3** ([33, Theorem 1.1]). *Let  $f : D \rightarrow E$ , where  $D \subseteq \{0, 1\}^N$ . Then  $Q_{0.01}(f) = O(\text{Adv}^\pm(f))$ .*

Combined with our result above, we obtain

$$B_{\epsilon, \delta}(f) = \Omega\left(\frac{Q_{0.01}(f)^2}{\epsilon}\right). \quad (89)$$

◀

## D Proof of Theorem 5.1

We restate and prove Theorem 5.1:

► **Theorem 5.1.** *Let  $f : D \rightarrow E$ , where  $D \subseteq \{0, 1\}^N$ . Suppose there is a classical randomized query algorithm  $\mathcal{A}$ , that makes at most  $T$  queries, and evaluates  $f$  with bounded error. Let the query results of  $\mathcal{A}$  on random seed  $s_{\mathcal{A}}$  be  $x_{p_1}, x_{p_2}, \dots, x_{p_{\tilde{T}(x)}}$ ,  $\tilde{T}(x) \leq T$ , where  $x$  is the hidden query string.*

Suppose there is another (not necessarily time-efficient) randomized algorithm  $\mathcal{G}$ , with random seed  $s_{\mathcal{G}}$ , which takes as input  $x_{p_1}, \dots, x_{p_{t-1}}$  and  $s_{\mathcal{A}}$ , and outputs a guess for the

<sup>5</sup> For boolean output (0 or 1) the  $2\delta$  term can be dropped, as we previously noted (Footnote 4).

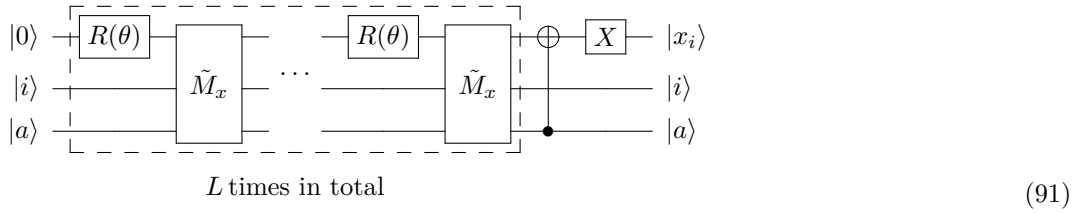
next query result  $x_{p_t}$  of  $\mathcal{A}$ . Assume that  $\mathcal{G}$  makes no more than an expected total of  $G$  mistakes (for all inputs  $x$ ). In other words,

$$\mathbf{E}_{s_{\mathcal{A}}, s_{\mathcal{G}}} \left\{ \sum_{t=1}^{\tilde{T}(x)} |\mathcal{G}(x_{p_1}, \dots, x_{p_{t-1}}, s_{\mathcal{A}}, s_{\mathcal{G}}) - x_{p_t}| \right\} \leq G \quad \forall x. \quad (90)$$

Note that  $\mathcal{G}$  is given the random seed  $s_{\mathcal{A}}$  of  $\mathcal{A}$ , so it can predict the next query index of  $\mathcal{A}$ . Then  $B_{\epsilon}(f) = O(TG/\epsilon)$ , and thus (by Theorem 4.1)  $Q(f) = O(\sqrt{TG})$ .

**Proof.** For the purposes of this proof, we use the characterization of  $B$  by the modified bomb construction given in section A.2. This proof is substantially similar to that of theorem 4.2.

The following circuit finds  $x_i$  with zero probability of explosion if  $x_i = a$ , and with an  $O(1/L)$  probability of explosion if  $x_i \neq a$  (in both cases the value of  $x_i$  found by the circuit is always correct):



where  $\theta = \pi/(2L)$  for some large number  $L$  to be picked later, and

$$R(\theta) = \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix} \quad (92)$$

The boxed part of the circuit is then simply  $[\tilde{M}_x(R(\theta) \otimes I \otimes I)]^L$ , applied to the state  $|0, i, a\rangle$ . We can analyze this circuit by breaking into cases:

- If  $x_i = a$ , then  $\tilde{M}_x|\psi\rangle|i, a\rangle = |\psi\rangle|i, a\rangle$  for any state  $|\psi\rangle$  in the control register. Thus the  $\tilde{M}_x$ 's act as identities, and the circuit simply applies the rotation  $R(\theta)^L = R(\pi/2)$  to the control register, rotating it from 0 to 1. We thus obtain the state  $|1, i, a\rangle$ ; the final CNOT and X gates add  $a \oplus 1 = x_i \oplus 1$  to the first register, giving  $|x_i, i, a\rangle$ .
- If  $x_i \neq a$ , then

$$\tilde{M}_x|0, i, a\rangle = |0, i, a\rangle, \quad \tilde{M}_x|1, i, a\rangle = 0 \quad (\text{for } x_i \neq a) \quad (93)$$

Therefore after each rotation  $R(\theta)$ , the projection  $\tilde{M}_x$  projects the control qubit back to 0:

$$\tilde{M}_x(R(\theta) \otimes I \otimes I)|0, i, a\rangle = \tilde{M}_x(\cos \theta|0\rangle + \sin \theta|1\rangle)|i, a\rangle = \cos \theta|0, i, a\rangle \quad (\text{for } x_i \neq a) \quad (94)$$

In this case the effect of  $\tilde{M}_x(R(\theta) \otimes I \otimes I)$  is to shrink the amplitude by  $\cos(\theta)$ ;  $L$  applications results in the state  $\cos^L(\theta)|0, i, a\rangle$ . The final CNOT and X gates add  $a \oplus 1 = x_i$  to the first register, giving  $|x_i, i, a\rangle$ .

The probability of explosion is 0 if  $x_i = a$ . If  $x_i \neq a$ , the probability of explosion is

$$1 - \cos^{2L} \left( \frac{\pi}{2L} \right) \leq \frac{\pi^2}{4L}. \quad (95)$$

Pick

$$L = \left\lceil \frac{\pi^2 G}{4\epsilon} \right\rceil. \quad (96)$$



Then the probability of explosion is 0 if  $x_i = a$ , and no more than  $\epsilon/G$  if  $x_i \neq a$ . If the bomb does not explode, then the circuit *always* finds the correct value of  $x_i$ .

We now construct the bomb query algorithm based on  $\mathcal{A}$  and  $\mathcal{G}$ . The bomb query algorithm follows  $\mathcal{A}$ , with each classical query replaced by the above construction. There are no more than  $TL \approx \pi^2 TG/(4\epsilon)$  bomb queries. At each classical query, we pick the guess  $a$  to be the guess provided by  $\mathcal{G}$ . The bomb only has a chance of exploding if the guess is incorrect; hence for all  $x$ , the total probability of explosion is no more than

$$\frac{\epsilon}{G} \mathbf{E}_{s_{\mathcal{A}}, s_{\mathcal{G}}} \left\{ \sum_{t=1}^{\hat{T}(x)} |\mathcal{G}(x_{p_1}, \dots, x_{p_{t-1}}, s_{\mathcal{A}}, s_{\mathcal{G}}) - x_{p_t}| \right\} \leq \epsilon \tag{97}$$

Thus replacing the classical queries of  $\mathcal{A}$  with our construction gives a bomb query algorithm with probability of explosion no more than  $\epsilon$ ; aside from the probability of explosion, this bomb algorithm makes no extra error over the classical algorithm  $\mathcal{A}$ . The number of queries this algorithm uses is

$$\tilde{B}_{\epsilon, \delta + \epsilon}(f) \leq \left\lceil \frac{\pi^2 G}{4\epsilon} \right\rceil T, \tag{98}$$

where  $\delta$  is the error rate of the classical algorithm. Therefore by Lemma A.1 and Theorem 4.1,

$$B_{\epsilon}(f) = O(B_{\epsilon, \delta + \epsilon}(f)) = O(\tilde{B}_{\epsilon, \delta + \epsilon}(f)) = O(TG/\epsilon) \tag{99}$$

◀

### E Proof of Theorem 5.3

We restate and prove Theorem 5.3:

► **Theorem 5.3** (Finding the first marked element in a list). Suppose there is an ordered list of  $N$  elements, and each element is either marked or unmarked. Then there is a bounded-error quantum algorithm for finding the **first** marked element in the list, or determines that no marked elements exist, such that:

- If the first marked element is the  $d$ -th element of the list, then the algorithm uses an expected  $O(\sqrt{d})$  time and queries.
- If there are no marked elements, then the algorithm uses  $O(\sqrt{N})$  time and queries.

**Proof.** We give an algorithm that has the stated properties. We first recall a quantum algorithm for finding the minimum in a list of items:

► **Theorem E.1** ([18]). *Given a function  $g$  on a domain of  $N$  elements, there is a quantum algorithm that finds the minimum of  $g$  with expected  $O(\sqrt{N})$  time and evaluations of  $g$ , making  $\delta < 1/10$  error.*

We now give our algorithm for finding the first marked element in a list. For simplicity, assume that  $N$  is a power of 2 (i.e.  $\log_2 N$  is an integer).

► **Algorithm E.2.**

1. For  $\ell = 2^0, 2^1, 2^2, \dots, 2^{\log_2 N} = N$ :

- Find the first marked element within the first  $\ell$  elements, or determine no marked element exists. This can be done by defining

$$g(i) = \begin{cases} \infty & \text{if } i \text{ is unmarked} \\ i & \text{if } i \text{ is marked,} \end{cases} \quad (100)$$

and using Theorem E.1 to find the minimum of  $g$ . This takes  $O(\sqrt{\ell}) = O(\sqrt{d})$  queries and makes  $\delta < 1/10$  error for each  $\ell$ . If a marked element  $i^*$  is found, the algorithm outputs  $i^*$  and stops.

- If no marked element was found in Step 1, the algorithm decides that no marked element exists.

We now claim that Algorithm E.2 has the desired properties. Let us break into cases:

- If no marked items exist, then no marked item can possibly be found in Step 1, so the algorithm correctly determines that no marked items exist in Step 2. The number of queries used is

$$\sum_{i=0}^{\log_2 N} \sqrt{2^i} = O(\sqrt{N}) \quad (101)$$

as desired.

- Suppose the first marked item is the  $d$ -th item in the list. Then in Step 1(a), if  $\ell \geq d$ , there is at least a  $1 - \delta$  probability that the algorithm will detect that a marked item exists in the first  $\ell$  elements and stop the loop. Letting  $\alpha = \lceil \log_2 d \rceil$ , the total expected number of queries is thus

$$\begin{aligned} \sum_{i=0}^{\alpha-1} \sqrt{2^i} + \sum_{i=\alpha}^{\log_2 N} \delta^{i-\alpha} \sqrt{2^i} + O(\sqrt{d}) &\leq \frac{2^{\alpha/2} - 1}{\sqrt{2} - 1} + \sqrt{2^\alpha} \frac{1}{1 - \sqrt{2}\delta} + O(\sqrt{d}) \\ &= O(\sqrt{2^\alpha}) + O(\sqrt{d}) \\ &= O(\sqrt{d}). \end{aligned} \quad (102)$$

The probability of not finding the marked item at the first  $\ell \geq d$  is at most  $\delta$ , and thus the total error of the algorithm is bounded by  $\delta$ . ◀

## F Explicit quantum algorithm for Theorem 5.2

- **Algorithm F.1** (Simulating a classical query algorithm by a quantum one).

*Input.* Classical randomized algorithm  $\mathcal{A}$  that computes  $f$  with bounded error. Classical randomized algorithm  $\mathcal{G}$  that guesses queries of  $\mathcal{A}$ . Oracle  $O_x$  for the hidden string  $x$ .

*Output.*  $f(x)$  with bounded error.

The quantum algorithm proceeds by attempting to produce the list of queries and results that  $\mathcal{A}$  would have made. More precisely, given a randomly chosen random seed  $s_{\mathcal{A}}$ , the quantum algorithm outputs (with constant error) a list of pairs  $(p_1(x), x_{p_1(x)}), \dots, (p_{\tilde{T}(x)}(x), x_{p_{\tilde{T}(x)}(x)})$ . This list is such that on random seed  $s_{\mathcal{A}}$ , the  $i$ -th query algorithm of  $\mathcal{A}$  is made at the position  $p_i(x)$ , and the query result is  $x_{p_i(x)}$ . The quantum algorithm then determines the output of  $\mathcal{A}$  using this list.

The main idea for the algorithm is this: we first assume that the guesses made by  $\mathcal{G}$  are correct. By repeatedly feeding the output of  $\mathcal{G}$  back into  $\mathcal{A}$  and  $\mathcal{G}$ , we can obtain a list of query values for  $\mathcal{A}$  without any queries to the actual black box. We then search for the first deviation of the string  $x$  from the predictions of  $\mathcal{G}$ ; assuming the first deviation is the  $d_1$ -th query, by Theorem 5.3 the search takes  $O(\sqrt{d_1})$  queries (ignoring error for now). We then know that all the guesses made by  $\mathcal{G}$  are correct up to the  $(d_1 - 1)$ -th query, and incorrect for the  $d_1$ -th query.

With the corrected result of the first  $d_1$  queries, we now continue by assuming again the guesses made by  $\mathcal{G}$  are correct starting from the  $(d_1 + 1)$ -th query, and search for the location of the next deviation,  $d_2$ . This takes  $O(\sqrt{d_2 - d_1})$  queries; we then know that all the guesses made by  $\mathcal{G}$  are correct from the  $(d_1 + 1)$ -th to  $(d_2 - 1)$ -th query, and incorrect for the  $d_2$ -th one. Continuing in this manner, we eventually determine all query results of  $\mathcal{A}$  after an expected  $G$  iterations.

We proceed to spell out our algorithm. For the time being, we assume that the algorithm for Theorem 5.3 (i.e. Algorithm E.2) has no error and thus requires no error reduction.

1. Initialize random seeds  $s_{\mathcal{A}}$  and  $s_{\mathcal{G}}$  for  $\mathcal{A}$  and  $\mathcal{G}$ . We will simulate the behavior of  $\mathcal{A}$  and  $\mathcal{G}$  on these random seeds. Initialize  $d = 0$ .  $d$  is such that we have determined the values of all query results of  $\mathcal{A}$  up to the  $d$ -th query. Also initialize an empty list  $\mathcal{L}$  of query pairs.
2. Repeat until either all query results of  $\mathcal{A}$  are determined, or  $100G$  iterations of this loop have been executed:
  - a. Assuming that  $\mathcal{G}$  always guesses correctly starting from the  $(d + 1)$ -th query, compute from  $\mathcal{A}$  and  $\mathcal{G}$  a list of query positions  $p_{d+1}, p_{d+2}, \dots$  and results  $\tilde{a}_{d+1}, \tilde{a}_{d+2}, \dots$ . This requires no queries to the black box.
  - b. Using our algorithm for finding the first marked element (Theorem 5.3, Algorithm E.2), find the first index  $d^* > d$  such that the actual query result of  $\mathcal{A}$  differs from the guess by  $\mathcal{G}$ , i.e.  $x_{p_d} \neq \tilde{a}_d$ ; or return that no such  $d^*$  exists. This takes  $O(\sqrt{d^* - d})$  time in the former case, and  $O(\sqrt{T - d})$  time in the latter.
  - c. We break into cases:
    - i. If an index  $d^*$  was found in Step 2b, then the algorithm decides the next mistake made by  $\mathcal{G}$  is at position  $d^*$ . It thus adds the query pairs  $(p_{d+1}, \tilde{a}_{d+1}), \dots, (p_{d^*-1}, \tilde{a}_{d^*-1})$ , and the pair  $(p_{d^*}, 1 - \tilde{a}_{d^*})$ , to the list  $\mathcal{L}$ . Also set  $d = d^*$ .
    - ii. If no index  $d^*$  was found in Step 2b, the algorithm decides that all remaining guesses by  $\mathcal{G}$  are correct. Thus the query pairs  $(p_{d+1}, \tilde{a}_{d+1}), \dots, (p_{\tilde{T}(x)}, \tilde{a}_{\tilde{T}(x)})$  are added to  $\mathcal{L}$ , where  $\tilde{T}(x) \leq T$  is the number of queries made by  $\mathcal{A}$ .
3. If the algorithm found all query results of  $\mathcal{A}$  in  $100G$  iterations of step 2, use  $\mathcal{L}$  to calculate the output of  $\mathcal{A}$ ; otherwise the algorithm fails.

We now count the total number of queries. Suppose  $g \leq 100G$  is the number of iterations of Step 2; if all query results have been determined,  $g$  is the number of wrong guesses by  $\mathcal{G}$ . Say the list of  $d$ 's found is  $d_0 = 0, d_1, \dots, d_g$ . Let  $d_{g+1} = T$ . Step 2 is executed for  $g + 1$  times, and the total number of queries is

$$O\left(\sum_{i=1}^{g+1} \sqrt{d_i - d_{i-1}}\right) = O\left(\sqrt{Tg}\right) = O\left(\sqrt{TG}\right) \tag{103}$$

by the Cauchy-Schwarz inequality.

We now analyze the error in our algorithm. The first source of error is cutting off the loop in Step 2: by Markov's inequality, for at least 99% of random seeds  $s_{\mathcal{G}}, s_{\mathcal{G}}$ ,  $\mathcal{G}$  makes no more

than  $100G$  wrong guesses. For these random seeds all query results of  $\mathcal{A}$  are determined. Cutting off the loop thus gives at most 0.01 error.

The other source of error is the error of Algorithm E.2 used in Step 2b: we had assumed that it could be treated as zero-error, but we now remove this assumption. Assuming each iteration gives error  $\delta'$ , the total error accrued could be up to  $O(g\delta')$ . It seems as if we would need to set  $\delta' = O(1/G)$  for the total error to be constant, and thus gain an extra logarithmic factor in the query complexity.

However, in his paper for oracle identification [31], Kothari showed that multiple calls to Algorithm E.2 can be composed to obtain a bounded-error algorithm based on span programs without an extra logarithmic factor in the query complexity; refer to [31, Section 3] for details. Therefore we can replace the iterations of Step 2 with Kothari's span program construction and get a bounded error algorithm with complexity  $O(\sqrt{TG})$ .

## **G** Proof of Theorem 6.5

We restate and prove Theorem 6.5:

► **Theorem 6.5.** The quantum query complexity of maximum bipartite matching is  $O(n^{7/4})$  in the adjacency matrix model, where  $n$  is the number of vertices.

**Proof.** We apply Theorem 5.2 to a classical algorithm. Classically, this problem is solved in  $O(n^{5/2})$  time by the Hopcroft-Karp [23] algorithm (here  $n = |V|$ ). We summarize the algorithm as follows (this summary roughly follows that of [4]):

► **Algorithm G.1** (Hopcroft-Karp algorithm for maximum bipartite matching [23]).

1. Initialize an empty matching  $\mathcal{M}$ .  $\mathcal{M}$  is a matching that will be updated until it is maximum.
2. Repeat the following steps until  $\mathcal{M}$  is a maximum matching:
  - a. Define the *directed* graph  $H = (V', E')$  as follows:

$$\begin{aligned}
 V' &= X \cup Y \cup \{s, t\} \\
 E' &= \{(s, x) \mid x \in X, (x, y) \notin \mathcal{M} \text{ for all } y \in Y\} \\
 &\cup \{(x, y) \mid x \in X, y \in Y, (x, y) \in E, (x, y) \notin \mathcal{M}\} \\
 &\cup \{(y, x) \mid x \in X, y \in Y, (x, y) \in E, (x, y) \in \mathcal{M}\} \\
 &\cup \{(y, t) \mid y \in Y, (x, y) \notin \mathcal{M} \text{ for all } x \in X\}
 \end{aligned} \tag{104}$$

where  $s$  and  $t$  are two extra auxiliary vertices. Note that if  $(s, x_1, y_1, x_2, y_2, \dots, x_\ell, y_\ell, t)$  is a path in  $H$  from  $s$  to  $t$ , then  $x_i \in X$  and  $y_i \in Y$  for all  $i$ . Additionally, the edges (aside from the first and last) alternate from being in  $\mathcal{M}$  and not being in  $\mathcal{M}$ :  $(x_i, y_i) \notin \mathcal{M}$ ,  $(y_i, x_{i+1}) \in \mathcal{M}$ . Such a path is called an *augmenting path* in the literature.

We note that a query to the adjacency matrix of  $E'$  can be simulated by a query to the adjacency matrix of  $E$ .

- b. Using breadth-first search, in the graph  $H$ , find the distances of all vertices from  $s$ . Let the distance from  $s$  to  $t$  be  $2\ell + 1$ .
- c. Find a maximal set  $S$  of vertex-disjoint shortest paths from  $s$  to  $t$  in the graph  $H$ . In other words,  $S$  should be a list of paths from  $s$  to  $t$  such that each path has length  $2\ell + 1$ , and no pair of paths share vertices except for  $s$  and  $t$ . Moreover, all other shortest paths from  $s$  to  $t$  share at least one vertex (except for  $s$  and  $t$ ) with a path in  $S$ . We describe how to find such a maximal set in Algorithm G.2.

- d. If  $S$  is empty, the matching  $M$  is a maximum matching, and we terminate. Otherwise continue:
- e. Let  $(s, x_1, y_1, x_2, y_2, \dots, x_\ell, y_\ell, t)$  be a path in  $S$ . Remove the  $\ell - 1$  edges  $(x_{i+1}, y_i)$  from  $\mathcal{M}$ , and insert the  $\ell$  edges  $(x_i, y_i)$  into  $\mathcal{M}$ . This increases  $|\mathcal{M}|$  by 1. Repeat for all paths in  $S$ ; there are no conflicts since the paths in  $S$  are vertex-disjoint.

Once again, we omit the proof of correctness of this algorithm; the correctness is guaranteed by Berge’s Lemma [8], which states that a matching is maximum if there are no more augmenting paths for the matching. Moreover,  $O(\sqrt{n})$  iterations of Step 2 suffice [23].

We now describe how to find a maximal set of shortest-length augmenting paths in Step 2(c). This algorithm is essentially a modified version of depth-first search:

► **Algorithm G.2** (Finding a maximal set of vertex-disjoint shortest-length augmenting paths).

*Input.* The directed graph  $H$  defined in Algorithm G.1, as well as the distances  $d_v$  of all vertices  $v$  from  $s$  (calculated in Step 2(b) of Algorithm G.1).

1. Initialize a set of paths  $S := \emptyset$ , set of vertices  $R := \{s\}$ , and a stack<sup>6</sup> of vertices  $\mathcal{L} := (s)$ .  $\mathcal{L}$  contains the ordered list of vertices that we have begun, but not yet finished, processing.  $R$  is the set of vertices that we have processed.  $S$  is the set of vertex-disjoint shortest-length augmenting paths that we have found.
2. Repeat until  $\mathcal{L}$  is empty:
  - a. If the vertex in the front of  $\mathcal{L}$  is  $t$ , we have found a new vertex-disjoint path from  $s$  to  $t$ :
    - Trace the path from  $t$  back to  $s$  by removing elements from the front of  $\mathcal{L}$  until  $s$  is at the front. Add the corresponding path to  $S$ .
    - Start again from the beginning of Step 2.
  - b. Let  $v$  be the vertex in the front of  $\mathcal{L}$  (i.e. the vertex *last* added to, and still in,  $\mathcal{L}$ ). Recall the distance from  $s$  to  $v$  is  $d_v$ .
  - c. Find  $w$  such that  $w \notin R$ ,  $d_w = d_v + 1$ , and  $(v, w)$  (as an edge in  $H$ ) has not been queried in this algorithm. If no such vertex  $w$  exists, remove  $v$  from  $\mathcal{L}$  and start from the beginning of Step 2.
  - d. Query  $(v, w)$  on the graph  $H$ .
  - e. If  $(v, w)$  is an edge, add  $w$  to the *front* of  $\mathcal{L}$ . If  $w \neq t$ , add  $w$  to  $R$ .
3. Output  $S$ , the maximal set of vertex-disjoint shortest-length augmenting paths.

We now return to Algorithm G.1 and count  $T$  and  $G$ . There is obviously no need to query the same edge more than once, so  $T = O(n^2)$ . If the algorithm always guesses, on a query  $(v, w)$ , that there is no edge between  $(v, w)$ , then it makes at most  $G = O(n^{3/2})$  mistakes: in Step 2(b) there are at most  $O(n)$  mistakes (see the proof of Theorem 6.2), while in Step 2(c)/Algorithm G.2 there is at most one queried edge leading to each vertex aside from  $t$ , and edges leading to  $t$  can be computed without queries to the adjacency matrix of  $H$ . Since Step 2 is executed  $O(\sqrt{n})$  times, our counting follows.

Thus there is a quantum query algorithm with complexity  $Q = O(\sqrt{TG}) = O(n^{7/4})$ . ◀

---

<sup>6</sup> A stack is a data structure such that elements that are first inserted into the stack are removed last.

# A Polylogarithmic PRG for Degree 2 Threshold Functions in the Gaussian Setting

Daniel M. Kane

University of California, San Diego  
Department of Computer Science and Engineering / Department of Mathematics  
9500 Gilman Drive #0404  
La Jolla, CA 92023, USA  
dakane@ucsd.edu

---

## Abstract

We construct and analyze a new pseudorandom generator for degree 2 polynomial threshold functions with respect to the Gaussian measure. In particular, we obtain one whose seed length is polylogarithmic in both the dimension and the desired error, a substantial improvement over existing constructions.

Our generator is obtained as an appropriate weighted average of pseudorandom generators against read once branching programs. The analysis requires a number of ideas including a hybrid argument and a structural result that allows us to treat our degree 2 threshold function as a function of a number of linear polynomials and one approximately linear polynomial.

**1998 ACM Subject Classification** G.3 Probability and Statistics

**Keywords and phrases** polynomial threshold function, pseudorandom generator, Gaussian distribution

**Digital Object Identifier** 10.4230/LIPIcs.CCC.2015.567

## 1 Introduction

We say that a function  $f : \mathbb{R}^n \rightarrow \{+1, -1\}$  is a (degree- $d$ ) *polynomial threshold function* (PTF) if it is of the form  $f(x) = \text{sgn}(p(x))$  for  $p$  some (degree- $d$ ) polynomial in  $n$  variables. Polynomial threshold functions make up a natural class of Boolean functions and have applications to a number of fields of computer science such as circuit complexity [1], communication complexity [14] and learning theory [11].

In this paper, we study the question of pseudorandom generators (PRGs) for polynomial threshold functions of Gaussians (and in particular for  $d = 2$ ). In other words, we wish to find explicit functions  $F : \{0, 1\}^s \rightarrow \mathbb{R}^n$  so that for any degree-2 polynomial threshold function  $f$

$$|\mathbb{E}_{x \sim_u \{0,1\}^s}[f(F(x))] - \mathbb{E}_{X \sim \mathcal{G}^n}[f(X)]| < \epsilon.$$

We say that such an  $F$  is a pseudorandom generator of seed length  $s$  that fools degree- $d$  polynomial threshold functions with respect to the Gaussian distribution to within  $\epsilon$ . In this paper, we develop a generator with  $s$  polylogarithmic in  $n$  and  $\epsilon$  in the case when  $d = 2$ .

### 1.1 Previous Work

There have been a number of papers dealing with the question of finding pseudorandom generators for polynomial threshold functions with respect to the Gaussian distribution or the Bernoulli distribution (i.e. uniform over  $\{-1, 1\}^n$ ). Several early works in this area showed that polynomial threshold functions of various degrees could be fooled by arbitrary



© Daniel M. Kane;  
licensed under Creative Commons License CC-BY  
30th Conference on Computational Complexity (CCC'15).

Editor: David Zuckerman; pp. 567–581



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

■ **Table 1** Generators Based on Limited Independence.

Paper	Bernoulli/Gaussian	d	k
Diakonikolas, Gopalan, Jaiswal, Servedio, Viola [3]	Bernoulli	1	$O(\epsilon^{-2} \log^2(\epsilon^{-1}))$
Diakonikolas, Kane, Nelson [4]	Gaussian	1	$O(\epsilon^{-2})$
Diakonikolas, Kane, Nelson [4]	Both	2	$O(\epsilon^{-8})^1$
Kane [7]	Both	$d$	$O_d(\epsilon^{-2^{O(d)}})$

$k$ -wise independent families of Gaussian or Bernoulli random variables. It should be noted that a  $k$ -wise independent family of Bernoulli random variables can be generated from a seed of length  $O(k \log(n))$ . Although, any  $k$ -wise independent family of Gaussians will necessarily have infinite entropy, it is not hard to show that a simple discretization of these random variables leads to a generator of comparable seed length. These results on fooling polynomial threshold functions with  $k$ -independence are summarized in Table 1.

Unfortunately, it is not hard to exhibit  $k$ -wise independent families of Bernoulli or Gaussian random variables that fail to  $\epsilon$ -fool the class of degree- $d$  polynomial threshold functions for  $k = \Omega(d^2 \epsilon^{-2})$ , putting a limit on what can be obtained through mere  $k$ -independence.

There have also been a number of attempts to produce pseudorandom generators by using more structure than limited independence. In [12], Meka and Zuckerman develop a couple of such generators in the Bernoulli case. Firstly, they make use of pseudorandom generators against space bounded computation to produce a generator of seed length  $O(\log(n) + \log^2(\epsilon^{-1}))$  in the special case where  $d = 1$ . By piecing together several  $k$ -wise independent families, they produce a generator for arbitrary degree PTFs of seed length  $2^{O(d)} \log(n) \epsilon^{-8d-3}$ . In [10], the author develops an improved analysis of this generator allowing for a seed length as small as  $O_{c,d}(\log(n) \epsilon^{-11-c})$ . For the Gaussian case, the author developed a generator of seed length  $2^{O_c(d)} \log(n) \epsilon^{-4-c}$  in [9]. This generator was given essentially as an average several random variables each picked independently from a  $k$ -wise independent family of Gaussians. The analysis of this generator was also improved in [10], obtaining a seed length of  $O_{c,d}(\log(n) \epsilon^{-2-c})$ . Finally, in [8] it was shown that this could be improved further by taking an average with unequal weights, given seed length  $O_{c,d}(\epsilon^{-c})$  for arbitrary degree and  $\log(n) \exp(O(\log(1/\epsilon)^{2/3} \log \log(1/\epsilon)^{1/3}))$  for degree 2. For a summary of these results, see Table 2.

The bound in [8] came from showing that for  $Y$  a weak pseudorandom generator (and in particular one that fools low degree moments) that

$$\left| \mathbb{E}[f(X)] - \mathbb{E}[f(\sqrt{1-\epsilon^2}X + \epsilon Y)] \right| \ll \epsilon^k \quad (1)$$

for any  $k$ . This followed from an important structure theorem that said that any polynomial  $p$  could be decomposed in terms of other polynomials,  $q_i$  so that when the  $q_i$  were localized near a random location then with high probability they would all be approximately linear polynomials. It was then shown that a moment matching random variable could fool such functions of approximately linear polynomials with high fidelity.

The bottleneck in this analysis comes in the size of the decomposition described above. On the one hand, for  $d > 2$  the size of the decomposition described above could potentially

<sup>1</sup> The bound in [4] for the Bernoulli case is actually  $\tilde{O}(\epsilon^{-9})$ , but this can be easily improved to  $O(\epsilon^{-8})$  using technology from [10].



■ **Table 2** Other Generators.

Paper	Bernoulli/Gaussian	d	s
Meka, Zuckerman [12]	Bernoulli	1	$O(\log(n) + \log^2(1/\epsilon))$
Kane [8]	Gaussian	1	$O(\log(n) + \log^{3/2}(1/\epsilon))$
Meka, Zuckerman [12]	Bernoulli	$d$	$\log(n)2^{O(d)}\epsilon^{-8d-3}$
Kane [9]	Gaussian	$d$	$\log(n)2^{O(d)}\epsilon^{-4.1}$
Kane [10]	Gaussian	$d$	$\log(n)O_d(\epsilon^{-2.1})$
Kane [10]	Bernoulli	$d$	$\log(n)O_d(\epsilon^{-11.1})$
Kane [8]	Gaussian	2	$\log(n) \exp(O(\log(1/\epsilon)^{2/3} \log \log(1/\epsilon)^{1/3}))$
Kane [8]	Gaussian	$d$	$\log(n)O_{c,d}(\epsilon^{-c})$
Kane, this paper	Gaussian	2	$O(\log^6(\epsilon) \log(n) \log \log(n/\epsilon))$

be quite large, though for  $d = 2$ , it can be handled explicitly. On the other hand, the implied constant in the approximation above depends exponentially on the size of this decomposition. While, we still do not know how to solve the former problem when  $d > 2$ , we can solve the latter in the case of degree-2 polynomial threshold functions.

In the special case of degree 2 functions, we end up with a decomposition of our quadratic polynomial as a function of a single approximately linear quadratic and several other linear polynomials. Fortunately, as discovered by Meka and Zuckerman, pseudorandom generators against read once branching programs are excellent at fooling linear polynomials (or even small numbers of them). As such generators also approximately fool the expectation of low degree polynomials (which is required for dealing with the approximately linear quadratic), they will actually be much better suited as our  $Y$  above. In fact, we can produce a pseudorandom generator for degree 2 polynomial threshold functions with polylogarithmic seed length. In particular, given an appropriate notion of a discretized Gaussian (the  $\delta$ -approximate Gaussian defined in Section 3), we have the following Theorem:

► **Theorem 1.1.** *Let  $\epsilon > 0$  and  $n$  a positive integer. For sufficiently large constant  $C$ , let  $\delta = \log(\epsilon)/C$  and  $\ell$  an integer at least  $\delta^{-3} \log(\epsilon)$ . For  $1 \leq i \leq \ell$  let  $Y_i$  be a family of  $n \exp(-\delta^{-1} \log(n/\delta))$ -approximate Gaussians seeded by a pseudorandom generator that fools read once branching programs of width  $\delta^{-2} \log(n/\delta)$  to within error  $\exp(-\delta^{-1} \log(n/\delta))$ . Let*

$$Y = \frac{\sum_{i=1}^{\ell} (1 - \delta^3)^{(\ell-1)/2} Y_i}{\sqrt{\sum_{i=1}^{\ell} (1 - \delta^3)^{\ell-1}}},$$

and let  $X$  be an  $n$  dimensional standard Gaussian. Then for any degree 2 polynomial threshold function  $f$  in  $n$  variables,

$$|\mathbb{E}[f(X)] - \mathbb{E}[f(Y)]| \leq \epsilon.$$

Furthermore, such  $Y$  can be constructed from generators of seed length of at most  $O(\log(\epsilon)^6 \log(n) \log \log(n/\epsilon))$ .

In Section 2, we will go over some basic notation and results. In Section 3, we introduce the concept of an approximate Gaussian, and show that families of them seeded by a PRG for read once branching programs will fool certain functions depending on a finite numbers of linear threshold functions and polynomials of low degree. In Section 4, we will prove our generalization of Equation (1). Finally, in Section 5, we will use this result to finish up our analysis and prove Theorem 1.1.

## 2 Background Information

### 2.1 Conventions

Throughout the paper we will use  $X, X_i, \dots$  as standard Gaussian random variables. We will usually use  $Y, Y_i, \dots$  to denote some sort of pseudorandom Gaussian.

### 2.2 Distribution of Values of Polynomials

Given a polynomial,  $p$ , we will need to know some basic information about how its values at random Gaussian inputs are distributed. Perhaps the most basic measure of such distribution is the average size of  $p(X)$ . In order to keep track this, we will make use of the  $L^t$  (and especially  $L^2$ ) norms. In particular, recall:

► **Definition 2.1.** If  $p : \mathbb{R}^n \rightarrow \mathbb{R}$  and  $t \geq 1$  then

$$|p|_t := (\mathbb{E}[|p(X)|^t])^{1/t}$$

where  $X$  is a standard Gaussian.

We will also need an anticoncentration result. That is a result telling us that the value of  $p(X)$  is unlikely to lie in any small neighborhood. In particular, we have:

► **Lemma 2.2** (Carbery and Wright, [2]). *If  $p$  is a degree- $d$  polynomial then*

$$\Pr(|p(X)| \leq \epsilon |p|_2) = O(d\epsilon^{1/d}).$$

Where the probability is over  $X$ , a standard  $n$ -dimensional Gaussian.

We will also need a concentration result for the values. To obtain one, we make use of the hypercontractive inequality below. The proof follows from Theorem 2 of [13].

► **Lemma 2.3.** *If  $p$  is a degree- $d$  polynomial and  $t > 2$ , then*

$$|p|_t \leq \sqrt{t-1}^d |p|_2.$$

This bound on higher moments allows us to prove a concentration bound on the distribution of  $p(X)$ . The following result is a well-known consequence that can be found, for example, in [6].

► **Corollary 2.4.** *If  $p$  is a degree- $d$  polynomial and  $N > 0$ , then*

$$\Pr_X(|p(X)| > N |p|_2) = O\left(2^{-(N/2)^{2/d}}\right).$$

**Proof.** Apply the Markov inequality and Lemma 2.3 with  $t = (N/2)^{2/d}$ . ◀

### 2.3 Hermite Polynomials

Recall that the Hermite polynomials  $h_a$  are an orthogonal set of polynomials with respect to the Gaussian distribution obtained by taking products of univariate Hermite polynomials in different coordinates. In particular,

$$\mathbb{E}[h_a(X)h_b(X)] = \delta_{a,b}.$$

We will need to make use of a few standard facts about the Hermite polynomials:

- Any degree- $d$  polynomial,  $p$ , can be written as a linear combination of Hermite polynomials of degree at most  $d$  so that the sum of the squares of the coefficients is  $|p|_2^2$  (and thus, the sum of the absolute values of the coefficients is at most  $n^d|p|_2$ ).
- A Hermite polynomial of degree  $d$  depends on at most  $d$  coordinates of its input. In fact it can be written as a product of one variable polynomials on these inputs.
- The sum of the absolute values of the coefficients of a Hermite polynomial of degree  $d$  is  $O(1)^d$ .

These properties are all easy to verify given basic facts about univariate Hermite polynomials.

### 3 Approximate Gaussians and Read Once Branching Programs

In order to produce a pseudorandom generator supported on a discrete set, we will first need to come up with a discrete version of the single variable Gaussian distribution. We will make use of the following notation:

► **Definition 3.1.** We say that a random variable  $Y$  is a  $\delta$ -approximate Gaussian, if there is a (correlated) standard (1-dimensional) Gaussian variable  $X$  so that

$$\Pr(|X - Y| > \delta) < \delta,$$

and  $|Y| = O(\log(\delta))$  with probability 1.

In particular, it is not difficult to generate a random variable with this property.

► **Lemma 3.2.** *There exists an explicit  $\delta$ -approximate Gaussian random variable that can be generated from a seed of length  $O(\log(\delta))$ .*

**Proof.** We assume that  $\delta$  is sufficiently small since otherwise there is nothing to prove. Let  $N = \lfloor \delta^{-3} \rfloor$ . Note that the random variable

$$X := \sqrt{-2 \log(z)} \cos(2\pi\theta)$$

is a random Gaussian if  $z$  and  $\theta$  independent uniform  $(0, 1)$  random variables. Let  $z'$  and  $\theta'$  be the roundings of  $z$  and  $\theta$  to the nearest half-integer multiple of  $1/N$ , and let

$$Y := \sqrt{-2 \log(z')} \cos(2\pi\theta').$$

Note that  $|z - z'|, |\theta - \theta'| \leq N^{-1}$ . From this it follows that

$$|X - Y| = O\left(\frac{1}{N \min(z, z', 1 - z, 1 - z')}\right).$$

Thus,  $|X - Y| < \delta$  with probability at least  $1 - \delta$ .

On the other hand,  $z'$  and  $\theta'$  are discrete uniform variables with  $O(\log(N)) = O(\log(\delta))$  bits of entropy each. Thus,  $Y$  can be generated from a seed of length  $O(\log(\delta))$ . ◀

We will also need to recall the concept of a read once branching program. An  $(M, D, n)$ -branching program is a program that is allowed to take only a single pass over an input consisting of  $n$   $D$ -bit blocks that is only allowed to save  $M$ -bits of memory between blocks. We will sometimes refer to this as a read once branching program of memory  $M$  (with  $n$  and  $D$  usually implicit). We note that there are small seed-length generators to fool such programs. In particular, we note the following theorem of [5]:

► **Theorem 3.3.** *There exists an explicit pseudorandom generator  $G$  with seed length  $O(M + d + \log(n/\epsilon) \log(n))$  so that if  $f$  is any Boolean function computed by an  $(M, D, n)$ -branching program, then*

$$|\mathbb{E}_{X \sim_u \{\{0,1\}^D\}^n} [f(X)] - \mathbb{E}[f(G)]| \leq \epsilon.$$

As shown in [12], using pseudorandom generators for read once branching programs is a good way to fool linear threshold functions, or by extension, things that depend on a small number of linear functions of the input. They will also fool the expectations of polynomials of low degree. An important building block for our construction will be families of approximate Gaussians seeded with a pseudorandom generator which fools read once branching programs. These, it turns out will simultaneously fool functions of a small number of linear functions and expectations of low degree polynomials in the following sense:

► **Proposition 3.4.** *Let  $s$  be a quadratic polynomial in  $n$  variables whose value depends on at most  $r$  linear polynomials. Let  $g(x)$  be the indicator function of the event that  $s(x)$  lies in  $I$  for some interval  $I$ . Let  $q(x)$  be a degree  $d$  polynomial in  $n$  variables. Let  $X$  be a standard Gaussian and let  $Y$  be a family on  $n$   $\delta_1$ -approximate Gaussians seeded by a PRG that fools read once branching programs of length  $n$  and memory  $M = O((d+r) \log(n/\delta_1))$  to error at most  $\delta_2$ . Then*

$$|\mathbb{E}[g(X)q(X)] - \mathbb{E}[g(Y)q(Y)]| \leq O(\log(\delta_1))^{d+1}(\delta_2 + n\delta_1^{1/4})n^d|q|_2.$$

First, we will need the following Lemma:

► **Lemma 3.5.** *Let  $s$  be a quadratic polynomial in  $n$  variables whose value depends on at most  $r$  linear polynomials. Let  $g(x)$  be the indicator function of the event that  $s(x)$  lies in  $I$  for some interval  $I$ . Let  $h(x)$  be a Hermite polynomial of degree  $d$ . Let  $X$  and  $Y$  be as given in Proposition 3.4. Then*

$$|\mathbb{E}[g(X)h(X)] - \mathbb{E}[g(Y)h(Y)]| \leq O(\log(\delta_1))^{d+1}(\delta_2 + n\delta_1^{1/4}).$$

**Proof.** We prove this in two steps. First, show that for  $Y'$  a family of  $n$  independent approximate Gaussians that  $\mathbb{E}[g(X)h(X)] \approx \mathbb{E}[g(Y')h(Y')]$ . This is because by correlating  $X$  and  $Y'$  appropriately, we can guarantee that  $X$  and  $Y'$  are close with high probability. This will mean that  $g(X) = g(Y')$  with high probability that  $h(X) \approx h(Y')$  with high probability. Next, we will need to show that  $\mathbb{E}[g(Y')h(Y')] \approx \mathbb{E}[g(Y)h(Y)]$ . This will hold because we can construct a read once branching program of small memory that computes approximations to the linear functions upon which  $s$  depends and the values of the (at most  $d$ ) coordinates upon which  $h$  depends.

We may assume that  $|s|_2 = 1$ . We begin by letting  $Y'$  be a family of independent  $\delta_1$ -approximate Gaussians. We can pick correlated copies of  $X$  and  $Y'$  so that with probability at least  $1 - n\delta_1$  each coordinate of  $X$  is within  $\delta_1$  of the corresponding coordinate of  $Y'$ . If this is the case, then  $|s(X) - s(Y')| = O(n \log(\delta_1)\delta_1)$ . By Lemma 2.2,  $s(X)$  is only within this distance of an endpoint of  $I$  with probability  $O(n^{1/2}\delta_1^{1/2} \log^d(\delta_1))$ . Thus, neglecting an event with this probability,  $g(X) = g(Y')$ . Let  $E$  be the event that  $g(X) \neq g(Y')$ , or that some coordinate of  $X$  and  $Y'$  differs by more than  $\delta_1$ . The contribution to  $\mathbb{E}[|g(X)h(X) - g(Y')h(Y')|]$  coming from times when  $E$  holds is at most

$$\mathbb{E}[\mathbf{1}_E(|h(X)| + |h(Y')|)],$$

which by Cauchy-Schwartz is at most

$$O((n^{1/4}\delta_1^{1/4} \log^{d/2}(\delta_1))\sqrt{\mathbb{E}[h(X)^2 + h(Y')^2]}) = O(n^{1/4}\delta_1^{1/4} \log^{d+1}(\delta_1)).$$

On the other hand  $\mathbb{E}[|h(X) - h(Y')|]$  when  $X$  and  $Y'$  agree to within  $\delta_1$  in each coordinate is  $O(n \log^d(\delta_1)\delta_1)$ . Thus,

$$|\mathbb{E}[g(X)h(X)] - \mathbb{E}[g(Y')h(Y')]| \leq O(\log^{d+1}(\delta_1)n\delta_1^{1/4}).$$

We now need to show that seeding  $Y'$  by a read once branching programs with  $M$  memory fools this expectation to within small error. Notice that a read once branching program with  $O((d+r)\log(n/\delta_1))$  memory can keep track of an approximation to within  $n^{-1}\delta_1^3$  of each of the  $r$  normalized linear functions that  $s$  depends on, and compute  $h$  to precision  $\delta_1$ . The latter is accomplished by writing  $h$  as  $\prod_{i=1}^n h_{a_i}(x_i)$  and keeping track of a running product  $\prod_{i=1}^m h_{a_i}(x_i)$  to relative precision  $\delta_1 O(\log(\delta_1))^{-d}(m/n)$ . This allows the program to compute the values of  $s$  and  $h$  to within an error of at most  $\delta_1$ .

Thus,  $\Pr(h(Y')g(Y') \geq c)$  is at most

$$\Pr(h(Y)g(Y) \geq c - \delta_1) + \Pr(s(Y') \text{ is within } \delta_1 \text{ of an endpoint of } I) + \delta_2.$$

Note that except for an event of probability  $n\delta_1$ , the difference between  $s(X)$  and  $s(Y')$  is at most  $O(n \log(\delta_1)\delta_1)$  and the former is this close to an endpoint of  $I$  with probability at most  $O(\log(\delta_1)\sqrt{n\delta_1})$ . Thus, with probability  $1 - O(\log(\delta_1)\sqrt{n\delta_1} + n\delta_1)$ ,  $s(Y')$  is not within  $\delta_1$  of a boundary of  $I$ . Thus for any  $c$ ,

$$\Pr(h(Y)g(Y) \geq c) \leq \Pr(h(Y')g(Y') \geq c - \delta_1) + O(\delta_2 + \log(\delta_1)n^{1/2}\delta_1^{1/2} + n\delta_1).$$

Integrating this over all  $|c| \leq O(\log(\delta_1))^d$  (which is the full range of values of  $h(Y')$  and  $h(Y)$ ), we find that

$$\mathbb{E}[g(Y)h(Y)] \leq \mathbb{E}[g(Y')h(Y')] + \delta_1 + O(\log(\delta_1))^{d+1}(\delta_2 + n\delta_1^{1/2}).$$

The lower bound follows similarly, and this completes the proof. ◀

**Proof of Proposition 3.4.** Note that we can write  $q$  as a linear combination of degree  $d$  hermite polynomials, where the sum of the absolute values of the coefficients is at most  $O(n^d|q|_2)$ . Our result follows from applying Lemma 3.5 to each term separately. ◀

We also note the following corollary when  $r = 0$ :

► **Corollary 3.6.** *Let  $X$  and  $Y$  be as in Proposition 3.4. Let  $q$  be a polynomial of degree at most  $d$  then*

$$|\mathbb{E}[q(X)] - \mathbb{E}[q(Y)]| \leq O(\log(\delta_1))^{d+1}(\delta_2 + n\delta_1^{1/4})n^d|q|_2.$$

## 4 The Key Result

Our analysis will depend heavily upon the following Proposition:

► **Proposition 4.1.** *Let  $\delta > 0$  and  $n$  a positive integer. Let  $C$  be a sufficiently large constant, and let  $Y$  be a family of  $n \exp(-C\delta^{-1} \log(n/\delta))$ -approximate Gaussians seeded by a pseudorandom generator that fools read once branching programs of memory  $C\delta^{-2} \log(n/\delta)$  to within error  $\exp(-C\delta^{-1} \log(n/\delta))$ . Let  $X$  be an  $n$  dimensional standard Gaussian. Then for any degree-2 polynomial threshold function  $f$  in  $n$  variables, we have that*

$$\left| \mathbb{E}[f(X)] - \mathbb{E}[f(\sqrt{1-\delta^3}X + \delta^{3/2}Y)] \right| = \exp(-\Omega(\delta^{-1})).$$

We first will need to show that this result holds for a certain class of quadratic polynomials. In particular, we define:

► **Definition 4.2.** A degree 2 polynomial  $p : \mathbb{R}^n \rightarrow \mathbb{R}$  is called  $(r, \delta)$ -approximately linear if it can be written in the form

$$p(x) = p_0(x \cdot v_1, \dots, x \cdot v_r) + x \cdot v + q(x)$$

for some vectors  $v_1, \dots, v_k, v$  with  $v$  orthogonal to  $v_i$ , and some degree-2 polynomials  $p_0$  and  $q$  so that  $|q|_2 < \delta|v|_2$ .

We now show an analogue of Proposition 4.1 for approximately linear polynomials:

► **Lemma 4.3.** Let  $k, r > 0$  be integers and  $\delta, \delta_1, \delta_2 > 0$  real numbers. Let  $p$  be an  $(r, \sqrt{\delta})$ -approximately linear polynomial in  $n$  variables with  $f$  the corresponding threshold function. Let  $X$  be an  $n$ -dimensional standard Gaussian, and  $Y$  a family on  $n$   $\delta_1$ -approximate Gaussians seeded by a PRG that fools read once branching programs of length  $n$  and memory  $M = C(k+r)\log(n/(\delta\delta_1\delta_2))$ , for sufficiently large  $C$ , to error at most  $\delta_2$ . Then

$$\left| \mathbb{E}[f(X)] - \mathbb{E}[f(\sqrt{1-\delta^2}X + \delta Y)] \right|$$

is at most

$$\leq \exp(-\Omega(\delta^{-1}))4^k + O(\log^{5k}(\delta_1)(\delta_2 + n\delta_1^{1/4}))O(nk)^{4k} + O(\delta k)^{2k} + O(2^{-k/2}).$$

The basic idea of the proof is as follows. First, we bin based on the approximate value of  $p_0$ . We are reduced to considering the expectation of the threshold function of a polynomial  $C + x \cdot v + q(x)$  times the indicator function of the event that  $p_0$  (a polynomial depending on a bounded number of linear functions) lies in a small interval. To deal with the threshold function, we note that averaging over possible values of  $X \cdot v$  smooths it out, and we may approximate it by its Taylor polynomial. Thus, we only need  $Y$  to fool the expectation of an indicator function of  $p_0$  lying in a small interval, times a low degree polynomial. This should hold by Proposition 3.4. The proof is as follows.

**Proof.** Since  $p$  is  $(r, \sqrt{\delta})$ -approximately linear, after rescaling we may assume that for some orthonormal set of vectors  $v, v_1, \dots, v_k$  that

$$p(x) = p_0(x \cdot v_1, \dots, x \cdot v_r) + x \cdot v + q(x)$$

for some quadratic polynomials  $p_0$  and  $q$  with  $|q|_2 < \sqrt{\delta}$ . We may assume that  $\delta \ll 1$ , for otherwise there is nothing to prove.

Let  $N = 2^k/|p|_2$ . Let  $I_n(x) := \mathbf{1}_{p_0(x) \in [n/N, (n+1)/N]}$  and let  $f_n(x) := I_n(x)f(x)$ . Let

$$f_n^+(x) = I_n(x)\text{sgn}(x \cdot v + q(x) + (n+1)/N), \text{ and } f_n^-(x) = I_n(x)\text{sgn}(x \cdot v + q(x) + (n)/N).$$

Note that  $f(x) = \sum_{n \in \mathbb{Z}} f_n(x)$ . Note also that  $f_n^+(x) \geq f_n(x) \geq f_n^-(x)$  for all  $x, n$ . We note that  $f_n^\pm(x)$  is actually a very close approximation to  $f_n(x)$ . In particular, by Lemma 2.2 if  $X$  is a random Gaussian then

$$\sum_{n \in \mathbb{Z}} \mathbb{E}[f_n^+(X) - f_n^-(X)] \leq \Pr(|p(X)| \leq 1/N) = O(2^{-k/2}).$$

Thus, it suffices to show that  $f_n^\pm(X)$  and  $f_n^\pm(\sqrt{1-\delta^2}X + \delta Y)$  have similar expectations for each  $n$ . To analyze this, let  $X_v$  be the component of  $X$  in the  $v$  direction, and  $X'$  be the component in the orthogonal directions. Let

$$\begin{aligned}
 g_n^\pm(X', Y) &:= \mathbb{E}_{X_v}[f_n^\pm(\sqrt{1-\delta^2}X + \delta Y)] \\
 &= I_n(X', Y)\mathbb{E}_{X_v}[\text{sgn}(C(X') + q_0(X', Y) + X_v(1 + q'_1(X') + q''_1(Y))) + X_v^2 q_2] \tag{2}
 \end{aligned}$$

where  $C(X')$  is a polynomial in  $X'$  and  $q_0, q'_1, q''_1$  and  $q_2$  are polynomials (of degree at most 2, 1, 1 and 0 respectively) of  $L^2$  norms at most  $|q_0|_2 = O(\delta)$ ,  $|q'_1|_2 = O(\sqrt{\delta})$ ,  $|q''_1|_2 = O(\delta)$ , and  $|q_2|_2 = O(\sqrt{\delta})$ . We may also assume that  $q_0$  is at most linear in the variables of  $X'$ , and that if we write  $q_0(X', Y) = \delta v \cdot Y + q'_0(X', Y)$ , then  $|q'_0(X', Y)|_2 = O(\delta^{3/2})$ . We claim that with probability  $1 - \exp(-\Omega(\delta^{-1}))$  over the choice of  $X'$  that the following hold:

1.  $\mathbb{E}_Y[q_0(X', Y)^2] = O(\delta^2)$ .
2.  $|q'_1(X')| < 1/3$ .

The first holds by Corollary 2.4 since  $\mathbb{E}_Y[q'_0(X', Y)^2]$  is a degree 2 polynomial in  $X'$  with  $L^2$  norm  $O(\delta^3)$ . Thus, with the desired probability  $\mathbb{E}_Y[q'_0(X', Y)^2] = O(\delta^2)$ , which implies the desired bound. The second holds by Corollary 2.4 since  $q'_1$  is a degree 1 polynomial with  $L^2$  norm  $O(\sqrt{\delta})$ . For the next part of the argument we will assume that we have fixed a value of  $X'$  so that the above holds.

Let  $q_1(X', Y) := q'_1(X') + q''_1(Y)$ . Note that if  $|q_0(X', Y)|, |q_1(X', Y)| < 2/3$ , then the polynomial  $C + q_0 + x(1 + q_1) + x^2 q_2$  cannot have more than one root with absolute value less than  $\Omega(\delta^{-1/2})$ . Since  $X_v$  cannot be larger than this except with probability  $\exp(-\Omega(\delta^{-1}))$ , the expectation above is  $\text{erf}(R) + \exp(-\Omega(\delta^{-1}))$ , where  $R$  is the smaller root of that quadratic. Furthermore, there will be no such root  $R$  unless  $|C| \ll \delta^{-1/2}$ . In such a case, by the quadratic formula, this root is

$$\begin{aligned}
 R &= \frac{-1 - q_1 + \sqrt{1 + 2q_1 + q_1^2 - 4q_2(C + q_0)}}{2q_2} \\
 &= (1 + q_1) \frac{\sqrt{1 - 4q_2(C + q_0)/(1 + q_1)^2} - 1}{2q_2} = \frac{C + q_0}{1 + q_1} + O(1). \tag{3}
 \end{aligned}$$

Thus, in the range  $|q_0|, |q_1| < 2/3$  and  $|C| \ll \delta^{-1/2}$  we have that the expectation in (2) is

$$\text{erf}(R) + \exp(-\Omega(\delta^{-1})).$$

Note that even for complex values of  $q_0$  and  $q_1$  with absolute value at most  $2/3$ , the  $\text{erf}(R)$  (with  $R$  given by Equation (3)) is complex analytic with absolute value uniformly bounded. Therefore, by Taylor expanding about  $q_0 = 0$  and  $q_1 = q'_1$ , we can find a polynomial  $P$  of degree at most  $2k$  (depending on  $q, C$  and  $X'$ ) so that  $\text{erf}(R)$  is

$$\begin{aligned}
 &P(q_0(X', Y), q_1(X', Y) - q'_1(X')) + O(q_0(X', Y))^{2k} + O(q_1(X', Y) - q'_1(Y))^{2k} \\
 &= P(q_0(X', Y), q''_1(Y)) + O(q_0(X', Y))^{2k} + O(q''_1(Y))^{2k}.
 \end{aligned}$$

Furthermore, the coefficients of  $P$  are all  $O(1)^k$ . The above must hold when  $|q_0|, |q''_1|$  are not at most  $1/3$ . On the other hand, this means that even when  $|q_0|, |q''_1|$  are larger than  $1/3$ , we have that  $P(q_0(X', Y), q''_1(X', Y)) \pm 1 = O(q_0(X', Y))^{2k} + O(q_1(X', Y))^{2k}$ . This means that the above formula holds for all values of  $q_0$  and  $q''_1$ . Thus,  $g_n^\pm(X', Y)$  is

$$G(X', Y) := \mathbf{1}_{s(X', Y) \in I}(P(q_0(X', Y), q''_1(Y)) + O(q_0(X', Y))^{2k} + O(q''_1(Y))^{2k}) + \exp(-\Omega(\delta^{-1}))$$



where  $s$  is some quadratic that depends on at most  $r$  linear functions,  $I$  is an interval. Thus,  $g(X', Y)$  will be approximately the product of an indicator function of something that depends on only a limited number linear functions of  $Y$  and a polynomial of bounded degree. Our proposition will hold essentially because PRGs for read once branching programs fool such functions as show in Proposition 3.4.

Note that  $P(q_0(Y), q_1''(Y))$  can be written as a polynomial of degree at most  $4k$  and  $L^2$  norm at most  $O(k)^{4k}$ . Letting  $G_0(y)$  be

$$G_0(y) := \mathbb{E}_X [\mathbf{1}_{s(X,y) \in I} P(q_0(X, y), q_1''(y))]$$

we have by Proposition 3.4 that

$$|\mathbb{E}[G_0(X)] - \mathbb{E}[G_0(Y)]| \leq O(\log^{5k}(\delta_1)(\delta_2 + n\delta_1^{1/4}))O(nk)^{4k}.$$

Similarly, if

$$G_1(y) := \mathbb{E}_X [\mathbf{1}_{s(X,y) \in I} (q_0(X, y)^{2k} + q_1''(X, y)^{2k})]$$

then

$$|\mathbb{E}[G_1(X)] - \mathbb{E}[G_1(Y)]| \leq O(\log^{5k}(\delta_1)(\delta_2 + n\delta_1^{1/4}))O(nk)^{4k}.$$

Also,

$$\mathbb{E}[G_1(X)] \leq O(\delta k)^{2k}$$

by Lemma 2.3. Therefore, we have that the difference in expectations between  $g_n^\pm(X', Y)$  and  $g_n^\pm(X', Z)$  where  $Z$  is an independent standard Gaussian, is at most

$$\exp(-\Omega(\delta^{-1})) + O(\log^{5k}(\delta_1)(\delta_2 + n\delta_1^{1/4}))O(nk)^{4k} + O(\delta k)^{2k}.$$

Thus,

$$\begin{aligned} & \left| \mathbb{E}[f_n^\pm(X)] - \mathbb{E}[f_n^\pm(\sqrt{1 - \delta^2}X + \delta Y)] \right| \\ & \leq \exp(-\Omega(\delta^{-1})) + O(\log^{5k}(\delta_1)(\delta_2 + n\delta_1^{1/4}))O(nk)^{4k} + O(\delta k)^{2k}. \end{aligned}$$

Therefore, we have that

$$\begin{aligned} & \sum_{|n| \leq 4^k} \left| \mathbb{E}[f_n(X)] - \mathbb{E}[f_n(\sqrt{1 - \delta^2}X + \delta Y)] \right| \\ & \leq \exp(-\Omega(\delta^{-1}))4^k + O(\log^{5k}(\delta_1)(\delta_2 + n\delta_1^{1/4}))O(nk)^{4k}\delta^{-k} + O(\delta k)^k \\ & \quad + \sum_n |\mathbb{E}[f_n^+(X) - f_n^-(X)]| \\ & \leq \exp(-\Omega(\delta^{-1}))4^k + O(\log^{5k}(\delta_1)(\delta_2 + n\delta_1^{1/4}))O(nk)^{4k}\delta^{-k} + O(\delta k)^k + O(2^{-k/2}). \end{aligned}$$

On the other hand,

$$\sum_{|n| \geq 4^k} \left| \mathbb{E}[f_n(X)] - \mathbb{E}[f_n(\sqrt{1 - \delta^2}X + \delta Y)] \right|$$

is at most the probability that either  $|p_0(X)|$  or  $|p_0(\sqrt{1 - \delta^2}X + \delta Y)|$  is more than  $2^k$  times the  $L^2$  norm of  $p$ , which is  $O(2^{-k})$  by the Markov bound and Corollary 3.6. Thus,

$$\begin{aligned} & \left| \mathbb{E}[f(X)] - \mathbb{E}[f(\sqrt{1 - \delta^2}X + \delta Y)] \right| \\ & \leq \sum_{|n| \in \mathbb{Z}} \left| \mathbb{E}[f_n(X)] - \mathbb{E}[f_n(\sqrt{1 - \delta^2}X + \delta Y)] \right| \\ & \leq \exp(-\Omega(\delta^{-1}))4^k + O(\log^{5k}(\delta_1)(\delta_2 + n\delta_1^{1/4}))O(nk)^{4k} + O(\delta k)^{2k} + O(2^{-k/2}). \end{aligned}$$

As desired.  $\blacktriangleleft$

We would like to reduce Proposition 4.1 to this case. Fortunately, it can be shown that after an appropriate random restriction that any quadratic polynomial can be made to be approximately linear with high probability.

► **Lemma 4.4.** *Let  $p$  be a degree 2 polynomial,  $\delta > 0$  and  $r$  a non-negative integer. Let  $X$  be a Gaussian random variable and  $p^{(X)}$  be the polynomial*

$$p^{(X)}(x) := p(\sqrt{1 - \delta^2}X + \delta x).$$

*Then with probability at least  $1 - \exp(-\Omega(r))$  over the choice of  $X$ ,  $p^{(X)}$  is  $(r, O(\delta))$ -approximately linear.*

**Proof.** For any polynomial  $q$ , let  $q^{(X)}$  be the polynomial

$$q^{(X)}(x) := q(\sqrt{1 - \delta^2}X + \delta x).$$

After diagonalizing the quadratic part of  $p$  and making an orthonormal change of variables we may write

$$p(x) = \sum_{i=1}^n p_i(x_i)$$

where  $p_i$  is a quadratic polynomial in one variable. Furthermore, we may assume that the quadratic term of  $p_i(x)$  is  $a_i x^2$  with  $|a_i|$  decreasing in  $i$ . Note that

$$p^{(X)}(x) = \sum_{i=1}^n p_i^{(X_i)}(x_i).$$

We may write  $p_i^{(X_i)}(x)$  as  $\delta^2 \sqrt{2} a_i h_2(x) + C_{i,1}(X_i)x + C_{i,0}(X_i)$  where  $h_2(x) = (x^2 - 1)/\sqrt{2}$  is the second Hermite polynomial, and  $C_{i,1}$  and  $C_{i,0}$  are appropriate constants depending on  $X_i$ . Note furthermore, that unless  $X_i$  lies within a small constant of the global maximum or minimum of  $p_i$  that  $|C_{i,1}(X_i)| = \Omega(\delta|a_i|)$ . Thus, with probability at least  $2/3$ , independently for each  $i$ , we have that  $|C_{i,1}(X_i)| = \Omega(\delta|a_i|)$ . Let  $I_i$  be the indicator random variable for the event that this happens.

From this it is easy to show that with probability  $1 - \exp(-\Omega(r))$  we have that  $\sum_{i=1}^m I_i \geq m/2 - r$  for all  $m$  (in fact the expected number of  $m$  for which this fails is exponentially small). We claim that if this occurs, then  $p^{(X)}$  is  $(r, O(\delta))$ -approximately linear. To show this, let  $S$  be the set of the  $r$  smallest indices  $i$  for which  $I_i = 0$ . We may write

$$p^{(X)}(x) = \left( \sum_{i \in S} p_i^{(X_i)}(x_i) + \sum_{i \notin S} C_{i,0}(X_i) \right) + \left( \sum_{i \notin S} C_{i,1}(X_i) e_i \right) \cdot X + \left( \sum_{i \notin S} \delta^2 \sqrt{2} a_i h_2(x_i) \right).$$

We claim that letting

$$p_0(x) = \sum_{i \in S} p_i^{(X_i)}(x_i) + \sum_{i \notin S} C_{i,0}(X_i), \quad v = \sum_{i \notin S} C_{i,1}(X_i) e_i, \quad q(x) = \sum_{i \notin S} \delta^2 \sqrt{2} a_i h_2(x_i)$$

shows that  $p^{(X)}$  is  $(r, O(\delta))$ -approximately linear.

It is clear that  $p_0$  depends on only the  $r$  linear functions  $x \cdot e_i$  for  $i \in S$ , that  $v$  is orthogonal to these  $e_i$ , and that  $p^{(X)}$  is the sum of  $p_0, x \cdot v$  and  $q$ . We have only to verify that  $|q|_2 = O(\delta)|v|$ . It is clear that  $|q|_2 = O(\delta^2) \sqrt{\sum_{i \notin S} a_i^2}$ . On the other hand, we have that

$$|v|_2 = \sqrt{\sum_{i \notin S} C_{i,1}^2(X_i)} \geq \Omega \left( \delta \sqrt{\sum_{i \notin S} I_i a_i^2} \right).$$

Thus, it suffices to show that

$$\sum_{i \notin S} I_i a_i^2 \geq \frac{1}{2} \sum_{i \notin S} a_i^2.$$

We can show this by Abel summation. In particular, for  $i \notin S$  let  $i'$  be the value of the next smallest integer not in  $S$  and let  $a_{n+1} = 0$ . We have that

$$\sum_{i \notin S} a_i^2 = \sum_{i \notin S} \sum_{j \notin S, j \geq i} a_j^2 - a_{j'}^2 = \sum_{j \notin S} (a_j^2 - a_{j'}^2) \left( \sum_{i \notin S, i \leq j} 1 \right).$$

Similarly,

$$\sum_{i \notin S} I_i a_i^2 = \sum_{i \notin S} I_i \sum_{j \notin S, j \geq i} I_j (a_j^2 - a_{j'}^2) = \sum_{j \notin S} (a_j^2 - a_{j'}^2) \left( \sum_{i \notin S, i \leq j} I_i \right).$$

On the other hand, for any  $j$  we have that

$$\sum_{i \notin S, i \leq j} I_i \geq \frac{1}{2} \sum_{i \notin S, i \leq j} 1.$$

Substituting into the above we find that

$$\sum_{i \notin S} I_i a_i^2 \geq \frac{1}{2} \sum_{i \notin S} a_i^2$$

and our result follows. ◀

Proposition 4.1 now follows easily by using Lemma 4.4 to reduce us to the case handled by Lemma 4.3.

**Proof.** Let  $f(x) = \text{sgn}(p(x))$  for some degree 2 polynomial  $p$ .

Let  $X_1$  and  $X_2$  be independent standard Gaussians. Note that

$$\mathbb{E}[f(\sqrt{1 - \delta^3}X + \delta^{3/2}Y)] = \mathbb{E}[f(\sqrt{1 - \delta}X_1 + \sqrt{\delta}(\sqrt{1 - \delta^2}X_2 + \delta Y))].$$

Let  $p^{(X_1)}$  be the polynomial given by

$$p^{(X_1)}(x) := p(\sqrt{1 - \delta}X_1 + \sqrt{\delta}x)$$

and let  $f^{(X_1)}(x) := \text{sgn}(p^{(X_1)}(x))$ . Note that

$$\mathbb{E}[f(\sqrt{1 - \delta^3}X + \delta^{3/2}Y)] = \mathbb{E}_{X_1}[\mathbb{E}_{X_2, Y}[f^{(X_1)}(\sqrt{1 - \delta^2}X_2 + \delta Y)]].$$

By Lemma 4.4, we have with probability  $1 - \exp(-\Omega(\delta^{-1}))$  over the choice of  $X_1$  that  $p^{(X_1)}$  is  $(\delta^{-1}, O(\sqrt{\delta}))$ -approximately linear. If this is the case, then by applying Lemma 4.3 with  $k$  a sufficiently small multiple of  $\delta^{-1}$ , we find that

$$\mathbb{E}_{X_2, Y}[f^{(X_1)}(\sqrt{1 - \delta^2}X_2 + \delta Y)] = \mathbb{E}[f^{(X_1)}(X)] + \exp(-\Omega(\delta^{-1})).$$

Putting these together, we find that

$$\begin{aligned} \mathbb{E}[f(\sqrt{1 - \delta^3}X + \delta^{3/2}Y)] &= \mathbb{E}_{X_1}[\mathbb{E}[f^{(X_1)}(X)]] + \exp(-\Omega(\delta^{-1})) \\ &= \mathbb{E}[f(\sqrt{1 - \delta}X_1 + \sqrt{\delta}X)] + \exp(-\Omega(\delta^{-1})) \\ &= \mathbb{E}[f(X)] + \exp(-\Omega(\delta^{-1})). \end{aligned}$$

◀

**5 Cleanup**

It is not difficult to complete the analysis of our generator given Proposition 4.1. We begin by applying Proposition 4.1 iteratively to obtain:

► **Lemma 5.1.** *Let  $\delta > 0$  and  $n, \ell$  be positive integers. Let  $C$  be a sufficiently large constant. For  $1 \leq i \leq \ell$  let  $Y_i$  be an independent copy of a family of  $n \exp(-C\delta^{-1} \log(n/\delta))$ -approximate Gaussians seeded by a pseudorandom generator that fools read once branching programs of memory  $C\delta^{-2} \log(n/\delta)$  to within error  $\exp(-C\delta^{-1} \log(n/\delta))$ . Let  $X$  be an  $n$  dimensional standard Gaussian. Then for any degree 2 polynomial threshold function  $f$  in  $n$  variables, we have that*

$$\left| \mathbb{E}[f(X)] - \mathbb{E} \left[ f \left( (1 - \delta^3)^{\ell/2} X + \delta^{3/2} \sum_{i=1}^{\ell} (1 - \delta^3)^{(\ell-1)/2} Y_i \right) \right] \right| \leq \ell \exp(-\Omega(\delta^{-1})).$$

**Proof.** The proof is by induction on  $\ell$ . The case of  $\ell = 0$  is trivial. Assuming that our Lemma holds for  $\ell$ , applying Proposition 4.1 to the threshold function

$$g(x) := f \left( (1 - \delta^3)^{\ell/2} x + \delta^{3/2} \sum_{i=1}^{\ell} (1 - \delta^3)^{(\ell-1)/2} Y_i \right),$$

we find that

$$\begin{aligned} & \mathbb{E} \left[ f \left( (1 - \delta^3)^{(\ell+1)/2} X + \delta^{3/2} \sum_{i=1}^{\ell+1} (1 - \delta^3)^{(\ell-1)/2} Y_i \right) \right] \\ &= \mathbb{E} \left[ f \left( (1 - \delta^3)^{\ell/2} X + \delta^{3/2} \sum_{i=1}^{\ell} (1 - \delta^3)^{(\ell-1)/2} Y_i \right) \right] + \exp(-\Omega(\delta^{-1})) \\ &= \mathbb{E}[f(X)] + (\ell + 1) \exp(-\Omega(\delta^{-1})). \end{aligned}$$

This completes the proof. ◀

Next, we note that when  $\ell$  is large, the coefficient of  $X$  above is small enough that it should have negligible probability of affecting the sign of the polynomial in question.

► **Lemma 5.2.** *Let  $\delta > 0$  and  $n, \ell$  be positive integers. Let  $C$  be a sufficiently large constant. For  $1 \leq i \leq \ell$  let  $Y_i$  be an independent copy of a family of  $n \exp(-C\delta^{-1} \log(n/\delta))$ -approximate Gaussians seeded by a pseudorandom generator that fools read once branching programs of memory  $C\delta^{-2} \log(n/\delta)$  to within error  $\exp(-C\delta^{-1} \log(n/\delta))$ . Let  $X$  be an  $n$  dimensional standard Gaussian. Then for any degree 2 polynomial threshold function  $f$  in  $n$  variables, we have that*

$$\left| \mathbb{E}[f(X)] - \mathbb{E} \left[ f \left( \frac{\sum_{i=1}^{\ell} (1 - \delta^3)^{(\ell-1)/2} Y_i}{\sqrt{\sum_{i=1}^{\ell} (1 - \delta^3)^{\ell-1}}} \right) \right] \right| \leq \ell \exp(-\Omega(\delta^{-1})) + O((1 - \delta^3)^{\ell/18}).$$

**Proof.** Let

$$Y := \frac{\sum_{i=1}^{\ell} (1 - \delta^3)^{(\ell-1)/2} Y_i}{\sqrt{\sum_{i=1}^{\ell} (1 - \delta^3)^{\ell-1}}},$$

and

$$Y' = (1 - \delta^3)^{\ell/2} X + \sqrt{1 - (1 - \delta^3)^{\ell}} Y.$$

By Lemma 5.1, it suffices to compare  $\mathbb{E}[f(Y)]$  with  $\mathbb{E}[f(Y')]$ . To do this, let  $p$  be the degree-2 polynomial defining the threshold function  $f$ . Consider

$$\mathbb{E} \left[ (p(Y) - p(Y'))^2 \right].$$

We may write this as  $\mathbb{E}[q(X, Y_1, \dots, Y_\ell)^2]$  for an appropriate quadratic polynomial  $q$ . Letting  $X_1, \dots, X_\ell$  be independent standard Gaussians, we have by repeated use of Corollary 3.6 that

$$\begin{aligned} \mathbb{E}[q(X, Y_1, \dots, Y_\ell)^2] &\leq (1 + \delta^5) \mathbb{E}[q(X, X_1, Y_2, \dots, Y_\ell)^2] \\ &\leq (1 + \delta^5)^2 \mathbb{E}[q(X, X_1, X_2, Y_3, \dots, Y_\ell)^2] \\ &\leq \dots \\ &\leq (1 + \delta^5)^\ell \mathbb{E}[q(X, X_1, \dots, X_\ell)^2] \\ &= (1 + \delta^5)^\ell \mathbb{E} \left[ \left( p(X) - p \left( (1 - \delta^3)^{\ell/2} X_1 + \sqrt{1 - (1 - \delta^3)^\ell} X \right) \right)^2 \right] \\ &= O((1 - \delta^3)^{\ell/3}) |p|_2^2. \end{aligned}$$

The factors of  $(1 + \delta^5)$  are showing up as a very loose approximation to the truth, and are obtained by noting that

$$\begin{aligned} &|\mathbb{E}[q(X, X_1, \dots, X_i, Y_{i+1}, \dots, Y_\ell)^2] - \mathbb{E}[q(X, X_1, \dots, X_{i-1}, Y_i, \dots, Y_\ell)^2]| \\ &\leq \exp(-\Omega(\delta^{-1})) \mathbb{E}_{X, X_j, Y_j, j \neq i} [\mathbb{E}[q(X, X_1, \dots, X_i, Y_{i+1}, \dots, Y_\ell)^4]^{1/2}] \\ &\leq \delta^5 \mathbb{E}[q(X, X_1, \dots, X_i, Y_{i+1}, \dots, Y_\ell)^2]. \end{aligned}$$

Let  $K = (1 - \delta^3)^{\ell/9} |p|_2$ . By Markov's inequality we have that  $|q(X, Y_i)| \leq K$  except with probability at most  $O((1 - \delta^3)^{\ell/18})$ . Let  $f_\pm(x) = \text{sgn}(p(x) \pm K)$ . By Lemma 2.2, we have that  $|\mathbb{E}[f_+(X)] - \mathbb{E}[f_-(X)]| \leq O(K^{1/2}) = O((1 - \delta^3)^{\ell/18})$ . By Lemma 5.1,  $|\mathbb{E}[f_\pm(X)] - \mathbb{E}[f_\pm(Y')]| \leq \ell \exp(-\Omega(\delta^{-1}))$ . On the other hand, with high probability  $|p(Y) - p(Y')| \leq K$  and thus with high probability

$$f_+(Y') \geq f(Y) \geq f_-(Y').$$

Therefore,

$$\begin{aligned} \mathbb{E}[f(Y)] &\leq \mathbb{E}[f_+(Y')] + O((1 - \delta^3)^{\ell/18}) \\ &\leq \mathbb{E}[f_+(X)] + O((1 - \delta^3)^{\ell/18}) + \ell \exp(-\Omega(\delta^{-1})) \\ &\leq \mathbb{E}[f(X)] + O((1 - \delta^3)^{\ell/18}) + \ell \exp(-\Omega(\delta^{-1})). \end{aligned}$$

The lower bound follows similarly, and this completes the proof.  $\blacktriangleleft$

Theorem 1.1 now follows immediately.

**Proof.** The result follows immediately from Lemma 5.2. We can obtain the stated seed length by using the generators from Lemma 3.2 and Theorem 3.3.  $\blacktriangleleft$

**Acknowledgements.** This research was done with the support of an NSF postdoctoral fellowship.

---

**References**

---

- 1 Richard Beigel *The polynomial method in circuit complexity*, Proc. of 8th Annual Structure in Complexity Theory Conference (1993), pp. 82–95.
- 2 A. Carbery, J. Wright *Distributional and  $L^q$  norm inequalities for polynomials over convex bodies in  $\mathbb{R}^n$*  Mathematical Research Letters, Vol. 8(3), pp. 233–248, 2001.
- 3 I. Diakonikolas, P. Gopalan, R. Jaiswal, R. Servedio, E. Viola, *Bounded Independence Fools Halfspaces* SIAM Journal on Computing, Vol. 39(8), pp. 3441–3462, 2010.
- 4 Ilias Diakonikolas, Daniel M. Kane, Jelani Nelson, *Bounded Independence Fools Degree-2 Threshold Functions*, Foundations of Computer Science (FOCS), 2010.
- 5 Russell Impagliazzo, Noam Nisan, Avi Wigderson *Pseudorandomness for network algorithms*, STOC (1994), pp. 356–364.
- 6 Svante Janson *Gaussian Hilbert Spaces*, Cambridge University Press, 1997.
- 7 Daniel M. Kane  *$k$ -Independent Gaussians Fool Polynomial Threshold Functions*, Conference on Computational Complexity (CCC), 2011.
- 8 Daniel M. Kane *A Pseudorandom Generator for Polynomial Threshold Functions of Gaussians with Subpolynomial Seed Length*, Conference on Computational Complexity (CCC) 2014.
- 9 Daniel M. Kane *A Small PRG for Polynomial Threshold Functions of Gaussians* Symposium on the Foundations Of Computer Science (FOCS), 2011.
- 10 Daniel M. Kane *A Structure Theorem for Poorly Anticoncentrated Gaussian Chaoses and Applications to the Study of Polynomial Threshold Functions*, manuscript <http://arxiv.org/abs/1204.0543>.
- 11 Adam R. Klivans, Rocco A. Servedio *Learning DNF in time  $2^{O(n^{1/3})}$* , J. Computer and System Sciences Vol. 68 (2004), pp. 303–318.
- 12 Raghu Meka, David Zuckerman *Pseudorandom generators for polynomial threshold functions*, Proceedings of the 42nd ACM Symposium on Theory Of Computing (STOC 2010).
- 13 Nelson *The free Markov field*, J. Func. Anal. Vol. 12(2), pp. 211–227, 1973.
- 14 Alexander A. Sherstov *Separating AC0 from depth-2 majority circuits*, SIAM J. Computing Vol. 38 (2009), pp. 2113–2129.

# Incompressible Functions, Relative-Error Extractors, and the Power of Nondeterministic Reductions (Extended Abstract)\*

Benny Applebaum<sup>1</sup>, Sergei Artemenko<sup>2</sup>, Ronen Shaltiel<sup>2</sup>, and Guang Yang<sup>3</sup>

- 1 Tel Aviv University, Tel Aviv, Israel, [bennyap@post.tau.ac.il](mailto:bennyap@post.tau.ac.il)
- 2 Haifa University, Haifa, Israel, [sartemen@gmail.com](mailto:sartemen@gmail.com), [ronen@cs.haifa.ac.il](mailto:ronen@cs.haifa.ac.il)
- 3 Tsinghua University, Beijing, China, [guang.research@gmail.com](mailto:guang.research@gmail.com)

---

## Abstract

A circuit  $C$  *compresses* a function  $f : \{0, 1\}^n \rightarrow \{0, 1\}^m$  if given an input  $x \in \{0, 1\}^n$  the circuit  $C$  can shrink  $x$  to a shorter  $\ell$ -bit string  $x'$  such that later, a computationally-unbounded solver  $D$  will be able to compute  $f(x)$  based on  $x'$ . In this paper we study the existence of functions which are *incompressible* by circuits of some fixed polynomial size  $s = n^c$ . Motivated by cryptographic applications, we focus on average-case  $(\ell, \epsilon)$  incompressibility, which guarantees that on a random input  $x \in \{0, 1\}^n$ , for every size  $s$  circuit  $C : \{0, 1\}^n \rightarrow \{0, 1\}^\ell$  and any unbounded solver  $D$ , the success probability  $\Pr_x[D(C(x)) = f(x)]$  is upper-bounded by  $2^{-m} + \epsilon$ . While this notion of incompressibility appeared in several works (e.g., Dubrov and Ishai [12]), so far no explicit constructions of efficiently computable incompressible functions were known. In this work we present the following results:

1. Assuming that  $\mathbf{E}$  is hard for exponential size nondeterministic circuits, we construct a polynomial time computable *boolean* function  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  which is incompressible by size  $n^c$  circuits with communication  $\ell = (1 - o(1)) \cdot n$  and error  $\epsilon = n^{-c}$ . Our technique generalizes to the case of PRGs against nonboolean circuits, improving and simplifying the previous construction of Shaltiel and Artemenko [5].
2. We show that it is possible to achieve *negligible* error parameter  $\epsilon = n^{-\omega(1)}$  for *nonboolean* functions. Specifically, assuming that  $\mathbf{E}$  is hard for exponential size  $\Sigma_3$ -circuits, we construct a nonboolean function  $f : \{0, 1\}^n \rightarrow \{0, 1\}^m$  which is incompressible by size  $n^c$  circuits with  $\ell = \Omega(n)$  and extremely small  $\epsilon = n^{-c} \cdot 2^{-m}$ . Our construction combines the techniques of Trevisan and Vadhan [47] with a new notion of *relative error* deterministic extractor which may be of independent interest.
3. We show that the task of constructing an incompressible *boolean* function  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  with *negligible* error parameter  $\epsilon$  cannot be achieved by “existing proof techniques”. Namely, *nondeterministic reductions* (or even  $\Sigma_i$  reductions) cannot get  $\epsilon = n^{-\omega(1)}$  for *boolean* incompressible functions. Our results also apply to constructions of standard Nisan-Wigderson type PRGs and (standard) boolean functions that are hard on average, explaining, in retrospective, the limitations of existing constructions. Our impossibility result builds on an approach of Shaltiel and Viola [40].

1998 ACM Subject Classification F.1.2 Modes of Computation

Keywords and phrases compression, pseudorandomness, extractors, nondeterministic reductions

Digital Object Identifier 10.4230/LIPIcs.CCC.2015.582

---

\* Full version appears in ECCC, TR15-051.





## 1 Introduction

In this paper we study several non-standard pseudorandom objects including incompressible functions, non-boolean PRGs and relative-error extractors for samplable and recognizable distributions. We present new constructions of these objects, relate them to each other and to standard pseudorandom objects, and study their limitations. Following some background on “traditional” pseudorandom objects (Section 1.1), we define and motivate incompressible functions, non-boolean PRGs and extractors for samplable distributions (Section 1.2). We continue with additional background on Hardness assumptions (Section 1.3), and state our results in Sections 1.4 – 1.7. The reader is referred to [1] for a full version of the paper.

### 1.1 Incomputable functions and Pseudorandom generators

Functions that are hard to compute on a random input, and pseudorandom generators (PRGs) are fundamental objects in Complexity Theory, Pseudorandomness and Cryptography.

► **Definition 1.1** (incomputable functions and pseudorandom generators).

- A function  $f : \{0, 1\}^n \rightarrow \{0, 1\}^m$  is **incomputable** by a class  $\mathcal{C}$  of functions if  $f$  is not contained in  $\mathcal{C}$ . We say that  $f$  is  $\epsilon$ -**incomputable** by  $\mathcal{C}$  if for every function  $C : \{0, 1\}^n \rightarrow \{0, 1\}^m$  in  $\mathcal{C}$ ,  $\Pr_{x \leftarrow U_n}[C(x) = f(x)] \leq \frac{1}{2^m} + \epsilon$ .
- A function  $G : \{0, 1\}^r \rightarrow \{0, 1\}^n$  is an  $\epsilon$ -**PRG** for a class  $\mathcal{C}$  of functions if for every function  $C : \{0, 1\}^n \rightarrow \{0, 1\}$  in  $\mathcal{C}$ ,  $|\Pr[C(G(U_r)) = 1] - \Pr[C(U_n) = 1]| \leq \epsilon$ .

A long line of research is devoted to achieving constructions of *explicit* incomputable functions and PRGs. As we are unable to give unconditional constructions of such explicit objects, the focus of many previous works is on achieving conditional constructions, that rely on as weak as possible unproven assumption. A common assumption under which explicit incomputable functions and PRGs can be constructed is the assumption below:

► **Assumption 1.2** (E is hard for exponential size circuits). *There exists a problem  $L$  in  $E = \text{DTIME}(2^{O(n)})$  and a constant  $\beta > 0$ , such that for every sufficiently large  $n$ , circuits of size  $2^{\beta n}$  fail to compute the characteristic function of  $L$  on inputs of length  $n$ .*

A long line of research in complexity theory is concerned with “hardness amplification” (namely, conditional constructions of explicit  $\epsilon$ -incomputable functions with small  $\epsilon$ ) and “hardness versus randomness tradeoffs” (namely, conditional constructions of explicit PRGs). We sum up some of the main achievements of this line of research in the theorem below.

► **Theorem 1.3** ([30, 34, 6, 25, 44]). *If  $E$  is hard for exponential size circuits, then for every constant  $c > 1$  there exists a constant  $a > 1$  such that for every sufficiently large  $n$ , and every  $r$  such that  $a \log n \leq r \leq n$ :*

- *There is a function  $f : \{0, 1\}^r \rightarrow \{0, 1\}$  that is  $n^{-c}$ -incomputable for size  $n^c$  circuits. Furthermore,  $f$  is computable in time  $\text{poly}(n^c)$ .<sup>1</sup>*
- *There is a function  $G : \{0, 1\}^r \rightarrow \{0, 1\}^n$  that is an  $n^{-c}$ -PRG for size  $n^c$  circuits. Furthermore,  $G$  is computable in time  $\text{poly}(n^c)$ .*

In the statement of Theorem 1.3 we allow input length  $r$  (of the functions  $f$  and  $G$ ) to vary between  $a \log n$  and  $n$ . It should be noted that the case of  $r > a \log n$  easily follows

<sup>1</sup> A statement like this means that we consider a family  $f = \{f_n\}$  for growing input lengths, and we think of  $r = r(n)$  as a function. We use this convention throughout the paper.

from the case of  $r = a \log n$ . We state the theorem this way, as we want to emphasize that by choosing  $r = n^{\Omega(1)}$ , we obtain incomputable functions/PRGs which run in time polynomial in their input length.

We also stress that in many settings in derandomization, increasing the input length  $r$  of a pseudorandom object, allows achieving very small error of  $\epsilon = 2^{-\Omega(r)}$ . In contrast, in Theorem 1.3 this dependence is not achieved. More precisely, if we set  $r = n^{\Omega(1)}$ , we only get  $\epsilon = n^{-c} = r^{-\Omega(1)}$  which is polynomially small in the input length. We will elaborate on this limitation later on.

## 1.2 Additional Pseudorandom objects

In this paper we consider generalizations of incomputable functions and PRGs that were introduced by Dubrov and Ishai [12]. We also consider the notion of extractors for samplable distributions introduced by Trevisan and Vadhan [47].

### 1.2.1 Incompressible functions

#### 1.2.1.1 Compression

Consider the following scenario. A computationally-bounded machine  $C$  wishes to compute some complicated function  $f$  on an input  $x$  of length  $n$ . While  $C$  cannot compute  $f(x)$  alone, it has a communication-limited access to a computationally-unbounded trusted “solver”  $D$ , who is willing to help. Hence,  $C$  would like to “compress” the  $n$ -bit input  $x$  to a shorter string  $x'$  of length  $\ell$  (the communication bound) while preserving the information needed to compute  $f(x)$ .

This notion of compression was introduced by Harnik and Naor [24] who studied the case where  $f$  is an NP-hard function. (Similar notions were also studied by the Parameterized Complexity community, see [24] for references.) Following Dubrov and Ishai [12], we focus on a scaled-down version of the problem where the gap between the complexity of  $f$  to the complexity of the compressor  $C$  is some fixed polynomial (e.g.,  $C$  runs in time  $n^2$ , while  $f$  is computable in time  $n^3$ ). In this setting, the notion of *incompressibility* is a natural strengthening of incomputability (as defined in Definition 1.1). We proceed with a formal definition. It is more useful to define the notion of “incompressibility” rather than “compressibility”. In the following, the reader should think of  $m < \ell < n$ .

► **Definition 1.4** (incompressible function [12]). A function  $f : \{0, 1\}^n \rightarrow \{0, 1\}^m$  is **incompressible** by a function  $C : \{0, 1\}^n \rightarrow \{0, 1\}^\ell$  if for every function  $D : \{0, 1\}^\ell \rightarrow \{0, 1\}^m$ , there exists  $x \in \{0, 1\}^m$  such that  $D(C(x)) \neq f(x)$ . We say that  $f$  is  **$\epsilon$ -incompressible** by  $C$  if for every function  $D : \{0, 1\}^\ell \rightarrow \{0, 1\}^m$ ,  $\Pr_{x \leftarrow U_n}[D(C(x)) = f(x)] \leq \frac{1}{2^m} + \epsilon$ . We say that  $f$  is  **$\ell$ -incompressible** (resp.  **$(\ell, \epsilon)$ -incompressible**) by a class  $\mathcal{C}$  if for every  $C : \{0, 1\}^n \rightarrow \{0, 1\}^\ell$  in  $\mathcal{C}$ ,  $f$  is incompressible (resp.  $\epsilon$ -incompressible) by  $C$ .

Incompressible functions are a generalization of incomputable functions in the sense that for every  $\ell \geq 1$  an  $(\ell, \epsilon)$ -incompressible function is in particular  $\epsilon$ -incomputable. However, incompressibility offers several additional advantages and yield some interesting positive and negative results.

#### 1.2.1.2 Communication lower-bounds for verifiable computation

As an immediate example, consider the problem of *verifiable computation* where a computationally bounded client  $C$  who holds an input  $x \in \{0, 1\}^n$  wishes to delegate the computation

of  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  (an  $n^3$ -time function) to a computationally strong (say  $n^{10}$ -time) untrusted server, while verifying that the answer is correct. This problem has attracted a considerable amount of research, and it was recently shown [28] that verifiable computation can be achieved with one-round of communication in which the client sends  $x$  to the server, and, in addition, the parties exchange at most polylogarithmic number of bits. If  $(1 - o(1)) \cdot n$ -incompressible functions exist, then this is essentially optimal. Furthermore, this lower-bound holds even in the preprocessing model ([15, 9, 2]) where the client is allowed to send long messages before seeing the input. Similar tight lower bounds can be shown for other related cryptographic tasks such as instance-hiding or garbled circuits (cf. [3, Section 6]).

### 1.2.1.3 Leakage-resilient storage [10]

On the positive side, consider the problem of storing a cryptographic key  $K$  on a computer that may leak information. Specifically, assume that our device was hacked by a computationally-bounded virus  $C$  who reads the memory and sends at most  $\ell$  bits to a (computationally unbounded) server  $D$ .<sup>2</sup> Is it possible to securely store a cryptographic key in such a scenario? Given an  $(\ell, \epsilon)$ -incompressible function  $f : \{0, 1\}^n \rightarrow \{0, 1\}^m$  we can solve the problem (with an information-theoretic security) by storing a random  $x \leftarrow \{0, 1\}^n$  and, whenever a cryptographic key  $K$  is needed, compute  $K = f(x)$  on-the-fly without storing it in the memory. For this application, we need average-case incompressibility (ideally with negligible  $\epsilon$ ), and a large output length  $m$ . Furthermore, it is useful to generalize incompressibility to the interactive setting in which the compressor  $C$  is allowed to have a multi-round interaction with the server  $D$ . (See the full version [1] for a formal definition.)

Unfortunately, so far no explicit constructions of incompressible functions (based on “standard assumptions”) are known, even in the worst-case setting.

### 1.2.2 PRGs for nonboolean circuits

Dubrov and Ishai [12] considered a generalization of pseudorandom generators, which should be secure even against distinguishers that output many bits. In the definition below, the reader should think of  $\ell \leq r < n$ .

► **Definition 1.5** (PRG for boolean and nonboolean distinguishers [12]). A function  $G : \{0, 1\}^r \rightarrow \{0, 1\}^n$  is an  $\epsilon$ -PRG for a function  $C : \{0, 1\}^n \rightarrow \{0, 1\}^\ell$  if the distributions  $C(G(U_r))$  and  $C(U_n)$  are  $\epsilon$ -close.<sup>3</sup>  $G$  is an  $(\ell, \epsilon)$ -PRG for a class  $\mathcal{C}$  of functions, if  $G$  is an  $\epsilon$ -PRG for every function  $C : \{0, 1\}^n \rightarrow \{0, 1\}^\ell$  in  $\mathcal{C}$ .

Indeed, note that a  $(1, \epsilon)$ -PRG is simply an  $\epsilon$ -PRG. Dubrov and Ishai noted that PRGs with large  $\ell$  can be used to reduce the randomness of sampling procedures. We now explain this application. In the definition below, the reader should think of  $\ell \leq n$ .

► **Definition 1.6** (Samplable distribution). We say that a distribution  $X$  on  $\ell$  bits is samplable by a class  $\mathcal{C}$  of functions  $C : \{0, 1\}^n \rightarrow \{0, 1\}^\ell$  if there exists a function  $C$  in the class such that  $X$  is  $C(U_n)$ .

Imagine that we can sample from some interesting distribution  $X$  on  $\ell = n^{1/10}$  bits using  $n$  random bits, by a procedure  $C$  that runs in time  $n^2$ . If we have a poly( $n$ )-time

<sup>2</sup> One may argue that if the outgoing communication is too large, the virus may be detected.

<sup>3</sup> We use  $U_n$  to denote the uniform distribution on  $n$  bits. Two distributions  $X, Y$  over the same domain are  $\epsilon$ -close if for any event  $A$ ,  $|\Pr[X \in A] - \Pr[Y \in A]| \leq \epsilon$ .

computable  $(\ell, \epsilon)$ -PRG  $G : \{0, 1\}^r \rightarrow \{0, 1\}^n$  against size  $n^2$  circuits, then the procedure  $P(s) = C(G(s))$  is a polynomial time procedure that samples a distribution that is  $\epsilon$ -close to  $X$  (meaning that even an unbounded adversary cannot distinguish between the two distributions). Furthermore, this procedure uses only  $r$  random bits (rather than  $n$  random bits) and we can hope to obtain  $r \ll n$ .

### 1.2.3 Extractors for samplable distributions

Deterministic (seedless) extractors are functions that extract randomness from “weak sources of randomness”. The reader is referred to [35, 36] for survey articles on randomness extractors.

► **Definition 1.7** (deterministic extractor). Let  $\mathcal{C}$  be a class of distributions over  $\{0, 1\}^n$ . A function  $E : \{0, 1\}^n \rightarrow \{0, 1\}^m$  is a  $(k, \epsilon)$ -extractor for  $\mathcal{C}$  if for every distribution  $X$  in the class  $\mathcal{C}$  such that  $H_\infty(X) \geq k$ ,  $E(X)$  is  $\epsilon$ -close to uniform.<sup>4</sup>

Trevisan and Vadhan [47] considered extractors for the class of distributions samplable by small circuits (e.g., distributions samplable by circuits of size  $n^2$ ).<sup>5</sup> The motivation presented by Trevisan and Vadhan is to extract randomness from “weak sources of randomness” in order to generate keys for cryptographic protocols. Indeed, extractors for samplable distributions are *seedless* and require no additional randomness (in contrast to seeded extractors). Note that for this application we would like extractors that run in polynomial time. The model of samplable distributions (say by circuits of size  $n^2$ ) is very general, and contains many subclasses of distributions studied in the literature on seedless extractors. Finally, Trevisan and Vadhan make the philosophical assumption that distributions obtained by nature must be efficiently samplable.

Summing up, if we are convinced that the physical device that is used by an honest party as a “weak source of randomness” has low complexity, (say size  $n^2$ ), then even an unbounded adversary that gets to *choose* or *affect* the source, cannot distinguish between the output of the extractor and the random string with advantage  $\geq \epsilon$ .

### 1.3 Hardness assumptions against nondeterministic and $\Sigma_i$ -circuits

In contrast to incomputable functions and (standard) PRGs,  $\text{poly}(n)$ -time constructions of the three objects above (incompressible functions, PRGs for nonboolean distinguishers and extractors for samplable distributions) are not known to follow from the assumption that E is hard for exponential size circuits. We now discuss stronger variants of this assumption under which such constructions can be achieved.

► **Definition 1.8** (nondeterministic circuits, oracle circuits and  $\Sigma_i$ -circuits). A *non-deterministic* circuit  $C$  has additional “nondeterministic input wires”. We say that the circuit  $C$  evaluates to 1 on  $x$  iff there exist an assignment to the nondeterministic input wires that makes  $C$  output 1 on  $x$ . An oracle circuit  $C^{(\cdot)}$  is a circuit which in addition to the standard gates uses an additional gate (which may have large fan in). When instantiated with a specific boolean function  $A$ ,  $C^A$  is the circuit in which the additional gate is  $A$ . Given a boolean function  $A(x)$ , an  $A$ -circuit is a circuit that is allowed to use  $A$  gates (in addition to the standard gates). An NP-circuit is a SAT-circuit (where SAT is the satisfiability function) a  $\Sigma_i$ -circuit

<sup>4</sup> For a distribution  $X$  over  $\{0, 1\}^n$ ,  $H_\infty(X) := \min_{x \in \{0, 1\}^n} \log \frac{1}{\Pr[X=x]}$ .

<sup>5</sup> In this paper we won't implicitly set a bound on the input length of the sampling circuit as such a bound is implied by the bound on its size.

is an  $A$ -circuit where  $A$  is the canonical  $\Sigma_i^P$ -complete language. The size of all circuits is the total number of wires and gates.<sup>6</sup>

Note, for example, that an NP-circuit is different than a nondeterministic circuit. The former is a nonuniform analogue of  $P^{\text{NP}}$  (which contains  $\text{coNP}$ ) while the latter is an analogue of NP. Hardness assumptions against nondeterministic/NP/ $\Sigma_i$  circuits appear in the literature in various contexts of complexity theory and derandomization [13, 29, 33, 47, 37, 16, 23, 38, 8, 39, 11]. Typically, the assumption used is identical to that of Assumption 1.2 except that “standard circuits” are replaced by one of the circuit types defined above. For completeness we restate this assumption precisely.

► **Definition 1.9.** We say that “E is hard for exponential size circuits of type X” if there exists a problem  $L$  in  $E = \text{DTIME}(2^{O(n)})$  and a constant  $\beta > 0$ , such that for every sufficiently large  $n$ , circuits of type X with size  $2^{\beta n}$  fail to compute the characteristic function of  $L$  on inputs of length  $n$ .

Such assumptions can be seen as the nonuniform and scaled-up versions of assumptions of the form  $\text{EXP} \neq \text{NP}$  or  $\text{EXP} \neq \Sigma_2^P$  (which are widely believed in complexity theory). As such, these assumptions are very strong, and yet plausible - the failure of one of these assumptions will force us to change our current view of the interplay between time, nonuniformity and nondeterminism.<sup>7</sup>

Hardness assumptions against nondeterministic or  $\Sigma_i$ -circuits appear in the literature in several contexts (most notably as assumptions under which  $\text{AM} = \text{NP}$ ). It is known that Theorem 1.3 extends to every type of circuits considered in Definition 1.8.

► **Theorem 1.10** ([25, 29, 37, 38]). *For every  $i \geq 0$ , the statement of Theorem 1.3 also holds if we replace every occurrence of the word “circuits” by “ $\Sigma_i$ -circuits” or alternatively by “nondeterministic  $\Sigma_i$ -circuits”.*

Thus, loosely speaking, if E is hard for exponential size circuits of type X, then for every  $c > 1$  we have PRGs and incomputable functions for size  $n^c$  circuits of type X, and these objects are  $\text{poly}(n^c)$ -time computable, and have error  $\epsilon = n^{-c}$ .<sup>8</sup>

## 1.4 New constructions based on hardness for nondeterministic circuits

Our first results are explicit constructions of incompressible functions and PRGs for non-boolean distinguishers from the assumption that E is hard for exponential size nondeterministic circuits.

<sup>6</sup> An alternative approach is to define using the Karp-Lipton notation for Turing machines with advice. For  $s \geq n$ , a size  $s^{\Theta(1)}$  deterministic circuit is equivalent to  $\text{DTIME}(s^{\Theta(1)})/s^{\Theta(1)}$ , a size  $s^{\Theta(1)}$  nondeterministic circuit is equivalent to  $\text{NTIME}(s^{\Theta(1)})/s^{\Theta(1)}$ , a size  $s^{\Theta(1)}$  NP-circuit is equivalent to  $\text{DTIME}^{\text{NP}}(s^{\Theta(1)})/s^{\Theta(1)}$ , a size  $s^{\Theta(1)}$  nondeterministic NP-circuit is equivalent to  $\text{NTIME}^{\text{NP}}(s^{\Theta(1)})/s^{\Theta(1)}$ , and a size  $s^{\Theta(1)}$   $\Sigma_i$ -circuit is equivalent to  $\text{DTIME}^{\Sigma_i^P}(s^{\Theta(1)})/s^{\Theta(1)}$ .

<sup>7</sup> Another advantage of constructions based on this type of assumptions is that any E-complete problem (and such problems are known) can be used to implement the constructions, and the correctness of the constructions (with that specific choice) follows from the assumption. We do not have to consider and evaluate various different candidate functions for the hardness assumption.

<sup>8</sup> Historically, the interest in PRGs for nondeterministic/NP circuits was motivated by the goal of proving that  $\text{AM} = \text{NP}$ , which indeed follows using sufficiently strong PRGs [29, 33, 37, 38]. It is important to note, that in contrast to PRGs against deterministic circuits, PRGs for nondeterministic circuits are trivially impossible to achieve, if the circuit can simulate the PRG. Indeed, this is why we consider PRGs against circuits of size  $n^c$  that are computable in larger time of  $\text{poly}(n^c)$ .

### 1.4.1 A construction of incompressible functions

Our first result is a construction of polynomial time computable incompressible functions, based on the assumption that E is hard for exponential size nondeterministic circuits. This is the first construction of incompressible functions from “standard assumptions”. The theorem below is stated so that the input length of the function is  $n$ . However, The input length can be shortened to any  $\Omega(\log n) \leq r \leq n$  as in the case of incomputable function stated in Theorem 1.3.

► **Theorem 1.11.** *If E is hard for exponential size nondeterministic circuits, then for every constant  $c > 1$  there exists a constant  $d > 1$  such that for every sufficiently large  $n$ , there is a function  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  that is  $(\ell, n^{-c})$ -incompressible for size  $n^c$  circuits, where  $\ell = n - d \cdot \log n$ . Furthermore,  $f$  is computable in time  $\text{poly}(n^c)$ .*

The theorem smoothly generalizes to the case of non-boolean functions  $f : \{0, 1\}^n \rightarrow \{0, 1\}^{n-\ell-d \log n}$ , and can also be extended to the interactive setting at the expense of strengthening the assumption to “E is hard for exponential size nondeterministic NP-circuits”. (See the full version [1].)

### 1.4.2 A construction of PRGs for nonboolean circuits

Dubrov and Ishai [12] showed that incompressible functions imply PRGs for nonboolean distinguishers. More precisely, they used the analysis of the Nisan-Wigderson generator [34] to argue that an incompressible function with the parameters obtained by Theorem 1.11 implies that for every constant  $c > 1$ , and every sufficiently large  $n$  and  $n^{\Omega(1)} \leq \ell < n$ , there is a  $\text{poly}(n^c)$ -time computable  $(\ell, n^{-c})$ -PRG  $G : \{0, 1\}^{r=O(\ell^2)} \rightarrow \{0, 1\}^n$  for circuits of size  $n^c$ . Using this relationship, one can obtain such PRGs under the assumption that E is hard for exponential size nondeterministic circuits. Note that a drawback of this result is that the seed length  $r$  is *quadratic* in  $\ell$ , whereas an optimal PRG can have seed length  $r = O(\ell)$ . This difference is significant in the application of reducing the randomness of sampling procedures (as explained in detail by Artemenko and Shaltiel [5]).

Artemenko and Shaltiel [5] constructed PRGs for nonboolean circuits with the parameters above, while also achieving seed length  $r = O(\ell)$ . However, they used the stronger assumption that E is hard for nondeterministic NP-circuits. In the theorem below we obtain the “best of both worlds”: We start from the assumption that E is hard for nondeterministic circuits and obtain PRGs with the optimal seed length of  $r = O(\ell)$ .

► **Theorem 1.12.** *If E is hard for exponential size non-deterministic circuits, then there exists a constant  $b > 1$  such that for every constant  $c > 1$  there exists a constant  $a > 1$  such that for every sufficiently large  $n$ , and every  $\ell$  such that  $a \log n \leq \ell \leq n$ , there is a function  $G : \{0, 1\}^{b \cdot \ell} \rightarrow \{0, 1\}^n$  that is an  $(\ell, n^{-c})$ -PRG for size  $n^c$  circuits. Furthermore,  $G$  is computable in time  $\text{poly}(n^c)$ .*

It should be noted that if  $\ell \leq c \log n$  then standard PRGs against size  $2 \cdot n^c$  circuits are also nb-PRGs. This is because any statistical test on  $\ell = c \log n$  bits can be implemented by a circuit of size  $n^c$ .

## 1.5 The power and limitations of nondeterministic reductions

### 1.5.1 Negligible error in pseudorandom objects?

A common theme in Theorems 1.3, 1.10, 1.11 and 1.12 is that we can get  $\epsilon = n^{-c}$ , but we never get  $\epsilon = n^{-\omega(1)}$  which would be desired, for example, for the virus application.



This holds even if we are allowed to increase the input/seed length  $r$ , and let  $r$  approach  $n$  (say  $r = n^{\Omega(1)}$ ). More generally, in all these results (and in fact, in all the literature on achieving incomputable functions/PRGs from the assumption that E is hard for exponential size *deterministic* circuits)  $1/\epsilon$  is always smaller than the running time of the constructed object. Consequently, polynomial time computable constructs do not obtain negligible error of  $\epsilon = n^{-\omega(1)}$ . This phenomenon is well understood, in the sense that there are general results showing that “current proof techniques” cannot beat this barrier. [40, 4]. (We give a more precise account of these results in the full version [1]).

However, there are examples in the literature where assuming hardness against *nondeterministic* (or more generally  $\Sigma_i$ ) circuits, it is possible to beat this barrier. The first example is the seminal work of Feige and Lund [13] on hardness of the permanent. More relevant to our setup are the following two results by Trevisan and Vadhan [47], and Drucker [11], stated precisely below. Note that in both cases, the target function is a polynomial time computable function that is  $\epsilon$ -incomputable for negligible  $\epsilon = n^{-\omega(1)}$ .

► **Theorem 1.13** (Nonboolean incomputable function with negligible error [47]). *If E is hard for exponential size NP-circuits, then there exists some constant  $\alpha > 0$  such that for every constant  $c > 1$  and for every sufficiently large  $n$ , there is a function  $f : \{0, 1\}^n \rightarrow \{0, 1\}^m$  that is  $\epsilon$ -incomputable by size  $n^c$  circuits for  $m = \alpha n$  and  $\epsilon = 2^{-(m/3)} = 2^{-\Omega(n)}$ . Furthermore,  $f$  is computable in time  $\text{poly}(n^c)$ .*

► **Theorem 1.14** (Nonboolean incomputable function with negligible error (corollary of [11])<sup>9</sup>). *For every  $c > 1$  there is a constant  $c' > c$  such that if there is a problem in P that for every sufficiently large  $n$  is  $(\frac{1}{2} - \frac{1}{n})$ -incomputable by nondeterministic circuits of size  $n^{c'}$ , then for every sufficiently large  $n$ , there is a function  $f : \{0, 1\}^n \rightarrow \{0, 1\}^{\sqrt{n}}$  that is  $\epsilon$ -incomputable by circuits of size  $n^c$ , for  $\epsilon = 2^{-n^{\Omega(1)}}$ . Furthermore,  $f$  is computable in time  $\text{poly}(n^c)$ .*<sup>10</sup>

It is important to note that in both cases above the target function that is constructed is *nonboolean*. We stress that the aforementioned lower bounds of [4] apply also to the case of nonboolean target functions, and the proofs above bypass these limitations by using *nondeterministic reductions*.

More precisely, assuming that the target function can be computed too well, the proofs need to contradict the assumption that E is hard for nondeterministic/ $\Sigma_i$ -circuits. They do this by designing a reduction. This reduction uses a deterministic circuit that computes the target function too well, in order to construct a nondeterministic/ $\Sigma_i$ -circuit that contradicts the assumption. This setting allows the reduction itself to be a nondeterministic/ $\Sigma_i$ -circuit. A precise definition of nondeterministic reductions appears in the full version [1].

Nondeterministic reductions are very powerful and previous limitations on reductions [40, 4] do not hold for nondeterministic reductions. (Indeed, Theorems 1.13 and 1.14 beat the barrier and achieve polynomial time computable functions that are  $n^{-\omega(1)}$ -incomputable).

<sup>9</sup> Drucker [11] considers a more general setting, on which we will not elaborate, and proves a direct product result. The result we state is a corollary that is easy to compare to the aforementioned results.

<sup>10</sup> The assumption of Theorem 1.14 is known to follow from the assumptions E is hard for exponential size nondeterministic circuits by Theorem 1.10. Consequently, the assumption used in Theorem 1.14 follows from the assumption in Theorem 1.13. The converse does not hold. We also remark that our Theorem 1.11 holds also if we replace the assumption by the following assumption that is similar in structure to Drucker’s assumption: For every  $c > 1$  there is a constant  $c' > c$  such that there is a problem in P that for every sufficiently large  $n$  is  $(\frac{1}{2} - \frac{1}{n})$ -incomputable by NP-circuits of size  $n^{c'}$ . The same holds for our Theorem 1.12 if we make the additional requirement that  $\ell = n^{\Omega(1)}$ .



Our Theorems 1.11 and 1.12 are also proven using nondeterministic reductions. This raises the question whether nondeterministic reductions can achieve error  $\epsilon = n^{-\omega(1)}$  in these cases. More generally, given the success of Trevisan and Vadhan, and Drucker, it is natural to hope that we can get  $\epsilon = n^{-\omega(1)}$  in the classical results stated in Theorem 1.3, if we are willing to assume the stronger assumption that E is hard for exponential size  $\Sigma_i$ -circuits, for some  $i > 0$ . Assuming this stronger assumption will allow the proof to use nondeterministic reductions (and the aforementioned lower bounds do not hold).

### 1.5.2 Limitations on nondeterministic reductions

In this paper we show that nondeterministic reductions (or even  $\Sigma_i$ -reductions) cannot be used to obtain a polynomial time  $n^{-\omega(1)}$ -incomputable *boolean* function, starting from the assumption that E is hard for exponential size  $\Sigma_i$ -circuits (no matter how large  $i$  is). To the best of our knowledge, our model of nondeterministic reduction (that is explained in the full version [1]) is sufficiently general to capture all known proofs in the literature on hardness amplification and PRGs.<sup>11</sup> This is a startling contrast between boolean and non-boolean hardness amplification - the latter can achieve negligible error, while the former cannot.<sup>12</sup> Our results provide a formal explanation for the phenomenon described above, and in particular, explains why Trevisan and Vadhan, and Drucker did not construct boolean functions.

We show that the same limitations hold, also for incompressible functions, PRGs against both boolean and nonboolean distinguishers, and extractors for samplable distributions. Our results are summarized informally below, and the precise statement of our limitations appears in the full version [1].

► **Informal Theorem 1.15.** *For every  $i \geq 0$  and  $c > 0$ , it is impossible to use “black-box reductions” to prove that the assumption that E is hard for  $\Sigma_i$ -circuits implies that for  $\epsilon = n^{-\omega(1)}$ , there is a  $\text{poly}(n)$ -time computable:*

- $\epsilon$ -incomputable functions  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  by size  $n^c$  circuits, or
- $\epsilon$ -PRG  $G : \{0, 1\}^r \rightarrow \{0, 1\}^n$  for size  $n^c$  circuits (the limitation holds for every  $r \leq n - 1$ ), or
- $(\ell, \epsilon)$ -PRG  $G : \{0, 1\}^r \rightarrow \{0, 1\}^n$  for size  $n^c$  circuits (the limitation holds for every  $r \leq n - 1$ ), or
- $(k, \epsilon)$ -extractor  $E : \{0, 1\}^n \rightarrow \{0, 1\}^m$  for size  $n^c$  circuits (the limitation holds for every  $m \geq 1$  and  $k \leq n - 1$ ).

*Furthermore, these limitations hold even if we allow reductions to perform  $\Sigma_i$ -computations, make adaptive queries to the “adversary breaking the security guarantee”, and receive arbitrary polynomial size nonuniform advice about the adversary.*

It is interesting to note that previous work on (deterministic) black-box reductions often cannot handle reductions that are both adaptive and nonuniform [20, 40] (see [4] for a discussion) and so the model of nondeterministic reductions that we consider is very strong.

<sup>11</sup> It should be noted that there are proof techniques (see e.g. [21, 22]) that bypass analogous limitations in a related setup. See [22] for a discussion.

<sup>12</sup> Another contrast between boolean and nonboolean hardness amplification was obtained by Shaltiel and Viola [40] for reductions that are non-adaptive constant depth circuits, and the reasons for the current contrast, are similar. Our proof follows the strategy of [40] as explained in detail in Section 2.

### 1.5.2.1 Related work on limitations on black-box hardness amplification

A “black-box proof of hardness amplification” consists of two components: A *construction* (showing how to compute the target function given access to the hardness assumption) and a *reduction* (showing that an adversary that is able to compute the target function too well, can be used to break the initial hardness assumption). We stress that in this paper we prove limitations on *reductions*. Our limitation holds without placing limitations on the complexity of the construction (and this only makes our results stronger). There is an orthogonal line of work which is interested in proving limitations on low complexity constructions. There is a superficial similarity to our work in that some of these results [48, 31, 32] show lower bounds on constructions implementable in the polynomial time hierarchy. However, this line of work is incomparable to ours, and is not relevant to the setting that we consider. Specifically, we want to capture cases in which the hardness assumption is for a function in exponential time. Typical polynomial time constructions use the hardness assumption on inputs of length  $O(\log n)$  where  $n$  is the input length of the target function (so that the initial function is computable in time polynomial in  $n$ ) and this allows the construction to inspect the entire truth table of the function in the hardness assumption. All previous limitations on the complexity of the *construction* trivially do not hold in this setting. We elaborate on our model and the meaning of our results in the full version [1].

## 1.6 Nonboolean incompressible functions with negligible error

In light of the previous discussion, if we want to achieve poly-time computable  $\epsilon$ -incompressible functions with  $\epsilon = n^{-\omega(1)}$  we must resort to nonboolean functions. In the next theorem we give such a construction.

► **Theorem 1.16** (Nonboolean incompressible function with negligible error). *If  $E$  is hard for exponential size  $\Sigma_3$ -circuits then there exists a constant  $\alpha > 0$  such that for every constant  $c > 1$  and every sufficiently large  $n$ , and  $m \leq \alpha \cdot n$  there is a function  $f : \{0, 1\}^n \rightarrow \{0, 1\}^m$  that is  $(\ell, n^{-c} \cdot 2^{-m})$ -incompressible for size  $n^c$  circuits, where  $\ell = \alpha \cdot n$ . Furthermore,  $f$  is computable in time  $\text{poly}(n^c)$ .*

We remark that the proof of Theorem 1.16 uses different techniques from the proof of Theorem 1.11. We also note that the conclusion of Theorem 1.16 is stronger than that of Theorems 1.13 and 1.14, even if we restrict our attention to  $\ell = 1$ . Specifically for  $m = \Omega(n)$ , we obtain that  $f : \{0, 1\}^n \rightarrow \{0, 1\}^{\Omega(n)}$  is  $\epsilon$ -incomputable by size  $n^c$  circuits, with  $\epsilon = n^{-c} \cdot 2^{-\Omega(n)}$ , meaning that circuits of size  $n^c$ , have probability at most  $\frac{1+n^{-c}}{2^m}$  of computing  $f(x)$ . This should be compared to the probability of random guessing which is  $\frac{1}{2^m}$ . Note that in the aforementioned theorems of [47, 11] the probability is larger than  $2^{-(m/2)}$  which is large compared to  $2^{-m}$ .

Moreover, the function we get is not only  $\epsilon$ -incomputable, but  $(\ell, \epsilon)$ -incompressible for large  $\ell = \Omega(n)$ , and we will show that this holds even in the interactive setting. Getting back to the memory leakage scenario, we will later see that (variants of) the theorem allows us to achieve a constant rate scheme (an  $m$  bit key is encoded by  $n = O(m)$  bits) which resists an  $n^c$ -time virus that (interactively) leaks a constant fraction of the stored bits.

## 1.7 Deterministic extractors with relative error

### 1.7.1 Previous work on extractors for samplable distributions

Trevisan and Vadhan constructed extractors for distributions samplable by size  $n^c$  circuits. The precise statement appears below.

► **Theorem 1.17** (Extractors for samplable distributions [47]). *If  $E$  is hard for exponential size  $\Sigma_4$ -circuits then there exists a constant  $\alpha > 0$  such that for every constant  $c > 1$  and sufficiently large  $n$ , and every  $m \leq \alpha n$  there is a  $((1 - \alpha) \cdot n, \frac{1}{n^c})$ -extractor  $E : \{0, 1\}^n \rightarrow \{0, 1\}^m$  for distributions samplable by size  $n^c$  circuits. Furthermore,  $E$  is computable in time  $\text{poly}(n^c)$ .<sup>13</sup>*

As explained earlier, our limitations explain why Trevisan and Vadhan did not achieve  $\epsilon = n^{-\omega(1)}$ . This may be a significant drawback in applications. In particular, if we use the extractor to generate keys for cryptographic protocols (as explained in Section 1.2.3) then it might be that an adversary that has a negligible probability of attacking the protocol under the uniform distribution, has a noticeable probability of attacking under the distribution output by the extractor.

### 1.7.2 Extractors with relative error

In order to circumvent this problem we suggest the following revised notion of statistical distance, and extractors.

► **Definition 1.18** (statistical distance with relative error). We say that a distribution  $Z$  on  $\{0, 1\}^m$  is  $\epsilon$ -close to uniform with relative error if for every event  $A \subseteq \{0, 1\}^m$ ,  $|\Pr[Z \in A] - \mu(A)| \leq \epsilon \cdot \mu(A)$  where  $\mu(A) = |A|/2^m$ .<sup>14</sup>

Note that if  $Z$  is  $\epsilon$ -close to uniform with relative error, then it is also  $\epsilon$ -close to uniform. However, we now also get that for every event  $A$ ,  $\Pr[Z \in A] \leq (1 + \epsilon) \cdot \mu(A)$  and this implies that events that are negligible under the uniform distributions cannot become noticeable under  $Z$ .

We now introduce a revised definition of deterministic extractors by replacing the requirement that the output is  $\epsilon$ -close to uniform by the requirement that the output is close to uniform with relative error.

► **Definition 1.19** (deterministic extractor with relative error). Let  $\mathcal{C}$  be a class of distributions over  $\{0, 1\}^n$ . A function  $E : \{0, 1\}^n \rightarrow \{0, 1\}^m$  is a  $(k, \epsilon)$ -relative-error extractor for  $\mathcal{C}$  if for every distribution  $X$  in the class  $\mathcal{C}$  such that  $H_\infty(X) \geq k$ ,  $E(X)$  is  $\epsilon$ -close to uniform with relative error.

To the best of our knowledge, this concept of “relative-error extractor” was not previously considered in the literature. We first observe that a standard probabilistic argument shows existence of such extractors for any small class of distributions. This follows by proving that random functions satisfy this property with high probability (using the same calculation as in the case of standard extractors). Moreover, this probabilistic argument works with random

<sup>13</sup>In [47], this is stated with  $m = 0.5 \cdot c \cdot \log n$ , but a more careful argument can give the stronger result that we state here. Another result that appears in [47] allows  $m$  to be  $(1 - \delta) \cdot n$  for an arbitrary constant  $\delta > 0$ , and then  $\Sigma_4$  is replaced by  $\Sigma_5$ ,  $\epsilon = 1/n$  and the running time is  $n^{b_{c,\delta}}$  for a constant  $b_{c,\delta}$  that depends only on  $c$  and  $\delta$ .

<sup>14</sup>While we'll use this definition mostly with  $\epsilon < 1$ , note that it makes sense also for  $\epsilon \geq 1$ .

$t$ -wise independent functions. Specifically, the following theorem was implicitly proven by Trevisan and Vadhan [47] (Proof of Proposition A.1):

► **Theorem 1.20** (Existence of relative-error extractors). *Let  $\mathcal{C}$  be a class of at most  $N$  distributions on  $\{0, 1\}^n$ . Then there exists a  $(k, \epsilon)$ -relative-error extractor  $E : \{0, 1\}^n \rightarrow \{0, 1\}^m$  for  $\mathcal{C}$  with  $m = k - 2\log(1/\epsilon) - O(\log \log N)$ . Furthermore, with probability at least  $1 - 2^{-n}$  a random  $O(n + \log N)$ -wise independent function  $h : \{0, 1\}^n \rightarrow \{0, 1\}^m$  is a  $(k, \epsilon)$ -relative-error extractor  $E : \{0, 1\}^n \rightarrow \{0, 1\}^m$  for  $\mathcal{C}$ .*

### 1.7.3 New constructions of relative error extractors for samplable distributions

We are able to extend Theorem 1.17 to hold with this new definition. Specifically:

► **Theorem 1.21** (Extractors for samplable distributions with relative error). *If  $E$  is hard for exponential size  $\Sigma_4$ -circuits then there exists a constant  $\alpha > 0$  such that for every constant  $c > 1$  and sufficiently large  $n$ , and every  $m \leq \alpha n$  there is a  $((1 - \alpha) \cdot n, \frac{1}{n^c})$ -relative-error extractor  $E : \{0, 1\}^n \rightarrow \{0, 1\}^m$  for distributions samplable by size  $n^c$  circuits. Furthermore,  $E$  is computable in time  $\text{poly}(n^c)$ .*

As previously explained this means that events that receive negligible probability under the uniform distribution also receive negligible probability under the output distribution of the extractor. We believe that this makes extractors for samplable distributions more suitable for cryptographic applications.

### 1.7.4 Relative error extractors for recognizable distributions

Shaltiel [41] introduced a notion of “recognizable distributions”.

► **Definition 1.22** (Recognizable distributions [41]). We say that a distribution  $X$  on  $n$  bits is **recognizable** by a class  $\mathcal{C}$  of functions  $C : \{0, 1\}^n \rightarrow \{0, 1\}$  if there exists a function  $C$  in the class such that  $X$  is uniform over  $\{x : C(x) = 1\}$ .

It is easy to see that extractors for distributions recognizable by small circuits translate into incompressible functions. Furthermore, relative-error extractors with large error translate into non-boolean incompressible functions with very small error.

► **Lemma 1.23.**

- An  $(n - (\ell + \log(1/\epsilon) + 1), \epsilon/2)$ -extractor for distributions recognizable by size  $n^c$  circuits, is an  $(\ell, \epsilon)$ -incompressible function for size  $n^c$  circuits.
- An  $(n - (\ell + \log(1/\epsilon) + m + 1), \epsilon/2)$  relative-error extractor  $f : \{0, 1\}^n \rightarrow \{0, 1\}^m$  for distributions recognizable by size  $n^c$  circuits, is an  $(\ell, \epsilon \cdot 2^{-m})$ -incompressible function for size  $n^c$  circuits.

This argument demonstrates (once again) the power of extractors with relative error. More precisely, note that even if  $\epsilon$  is noticeable, we get guarantees on probabilities that are negligible! This lemma shows that in order to construct nonboolean incompressible functions with very low error, it is sufficient to construct extractors for recognizable distributions with relative error that is noticeable.

This lemma follows because if we choose  $X \leftarrow U_n$  and consider the distribution of  $(X|C(X) = a)$  for some compressed value  $a \in \{0, 1\}^\ell$  that was computed by the compressor  $C$ , then this distribution is recognizable, and for most  $a$ , it has sufficiently large min-entropy

for the extractor  $f$ . It follows that  $f(X)$  is close to uniform with relative error even after seeing  $C(X)$ . However, in a distribution that is  $\epsilon$ -close to uniform with relative error, no string has probability larger than  $(1 + \epsilon) \cdot 2^{-m}$ , and so even an unbounded adversary that sees  $C(X)$  cannot predict  $f(X)$  with advantage better than  $\epsilon \cdot 2^{-m}$  over random guessing. We give a full proof in a more general setup in the formal section.

Our next result is a construction of a relative-error extractor for recognizable distributions.

► **Theorem 1.24** (Extractors for recognizable distributions with relative error). *If  $E$  is hard for exponential size  $\Sigma_3$ -circuits then there exists a constant  $\alpha > 0$  such that for every constant  $c > 1$  and sufficiently large  $n$ , and every  $m \leq \alpha n$  there is a  $((1 - \alpha) \cdot n, \frac{1}{n^c})$ -relative error extractor  $E : \{0, 1\}^n \rightarrow \{0, 1\}^m$  for distributions recognizable by size  $n^c$  circuits. Furthermore,  $E$  is computable in time  $\text{poly}(n^c)$ .*

### 1.7.4.1 Application in the leakage resilient scenario

The same reasoning applies in the memory leakage scenario described in Section 1.2.1. Using a relative error extractor for recognizable distributions  $f$ , we can achieve a constant rate scheme (an  $m$  bit key is encoded by  $n = O(m)$  bits) which resists an  $n^c$ -time virus who (interactively) leaks a constant fraction of the stored bits in the following strong sense: Say that the key  $K = f(x)$  is used as the key of some cryptographic scheme  $F_K$ , and that the scheme  $F_K$  is secure in the sense that the probability that an adversary breaks the scheme is negligible (under a uniform key), then the scheme remains secure even in the presence of the additional information that was released by the virus.

## 2 Overview and Technique

In this section we present a high level overview of the techniques used to prove our results.

### 2.1 Boolean incompressible functions with error $n^{-c}$

We start with an overview of the proof of Theorem 1.11. Our goal is to construct a boolean incompressible function for size  $n^c$  circuits. Consider a family of  $\text{poly}(n^c)$ -wise independent hash functions  $H = \{h_s : \{0, 1\}^n \rightarrow \{0, 1\}\}$ . We can sample from such a family using  $t = n^{O(c)}$  random bits. An easy counting argument (see e.g. [47]) shows that for every not too large class of distributions with min-entropy  $k$  (such as the class of distributions recognizable by size  $n^c$  circuits) a random  $h_s \leftarrow H$ , is with high probability an extractor for distributions in the class.

By Lemma 1.23, a random  $h \leftarrow H$  is w.h.p. an  $(\ell, \epsilon)$ -incompressible function for  $\ell = (1 - o(1)) \cdot n$  and negligible  $\epsilon$ . We are assuming that  $E$  is hard for exponential size nondeterministic circuits, and by Theorem 1.10, there is a  $\text{poly}(n^t)$ -time computable PRG  $G : \{0, 1\}^n \rightarrow \{0, 1\}^t$  for size  $n^{O(t)}$  nondeterministic circuits. We construct an incompressible function  $f : \{0, 1\}^{2n} \rightarrow \{0, 1\}$  as follows:

$$f(x, y) = h_{G(y)}(x)$$

Note that  $f$  is computable in polynomial time. In order to show that  $f$  is  $(\ell, n^{-c})$ -incompressible, it is sufficient to show that for  $(1 - n^{-c}/2)$ -fraction of seeds  $y \in \{0, 1\}^n$ ,  $f(y, \cdot) = h_{G(y)}(\cdot)$  is  $(\ell, n^{-c}/2)$ -incompressible.

We will show that for  $\epsilon = 1/\text{poly}(n)$ , there exists a polynomial size nondeterministic circuit  $P$ , that when given  $s \in \{0, 1\}^t$ , accepts if  $h_s$  is not  $(\ell, 2\epsilon)$ -incompressible, and rejects

if  $h_s$  is  $(\ell, \epsilon)$ -incompressible. A key observation is that as  $\text{AM} \subseteq \text{NP}/\text{poly}$ , it is sufficient to design an Arthur-Merlin protocols  $P$ , and furthermore by [7, 19] we can allow this protocol to be a private coin, constant round protocol, with small (but noticeable) gap between completeness and soundness.

We now present the protocol  $P$ : Merlin (who is claiming that  $h_s$  is not  $(\ell, 2\epsilon)$ -incompressible) sends a circuit  $C: \{0, 1\}^n \rightarrow \{0, 1\}^\ell$  of size  $n^c$  (which is supposed to compress the function well). Arthur, chooses private coins  $x \leftarrow U_n$ , and sends  $C(x)$  to Merlin. Merlin responds by guessing  $h_s(x)$ , and Arthur accepts if Merlin guessed correctly. It is immediate that this protocol has completeness  $\frac{1}{2} + 2\epsilon$  and soundness  $\frac{1}{2} + \epsilon$  and the gap is large enough to perform amplification.

It follows that for a uniform  $y$ , w.h.p.  $h_{G(y)}$  is  $2\epsilon$ -incompressible, as otherwise the nondeterministic circuit  $P$  distinguishes the output of  $G$  from uniform.<sup>15</sup>

We remark that this approach can be extended to yield nonboolean incompressible functions. However, using this approach we cannot get  $\epsilon = n^{-\omega(1)}$ . This is because the error of the final function  $f$  is at least the error of the PRG  $G$ , which cannot be negligible. We later present our construction of nonboolean incompressible function with very low error (as promised in Theorem 1.16), which works by giving a construction of relative error extractors for recognizable distributions (using quite different techniques).

This approach of explicit construction by using PRGs to derandomize a probabilistic construction was suggested in full generality by Klivans and van Melkebeek [29], and was used in many relevant works such as [38, 5]. However, the use of AM protocols with *private coins* enables us to come up with very simple proofs that improve upon previous work. An example is our next result that improves a recent construction of [5].

## 2.2 PRGs for nonboolean distinguishers

We now give an overview of the proof of Theorem 1.12 and show how to construct PRGs against nonboolean distinguishers. The argument is similar to that of the previous section. This time we take a  $\text{poly}(n^c)$ -wise independent family of hash functions  $H = \{h_s: \{0, 1\}^{2\ell} \rightarrow \{0, 1\}^n\}$ . We show that w.h.p. a random  $h_s \leftarrow H$  is an  $(\ell, \epsilon)$ -PRG with very small  $\epsilon$ . (This follows because by a standard calculation, w.h.p.  $h_s$  is a  $(\epsilon \cdot 2^{-\ell})$ -PRG for size  $n^c$ , and this easily implies that it is an  $(\ell, \epsilon)$ -PRG [5]). Our final PRG is again  $G'(x, y) = h_{G(y)}(x)$  for the same PRG  $G$  as in the previous section.

Following our earlier strategy, it is sufficient to design a constant round, private coin AM protocol  $P$  with noticeable gap  $\epsilon$  between completeness and soundness, such that given  $s \in \{0, 1\}^t$ ,  $P$  distinguishes the case that  $h_s$  is not an  $(\ell, 2\epsilon)$ -PRG from the case that  $h_s$  is an  $(\ell, \epsilon)$ -PRG.

We now present such a protocol, that is similar in spirit to the graph non-isomorphism protocol [18]. Merlin (who is claiming that  $h_s$  is not a good PRG) sends a circuit  $C: \{0, 1\}^n \rightarrow \{0, 1\}^\ell$  (that is supposed to distinguish the output of  $h_s$  from random). Arthur tosses a private fair coin, and either sends  $C(y)$  for  $y \leftarrow U_n$ , or  $C(h_s(x))$  for  $x \leftarrow U_{2\ell}$ ,

<sup>15</sup>Note that for this argument it is sufficient to have a PRG  $G: \{0, 1\}^n \rightarrow \{0, 1\}^{t=n^{O(\epsilon)}}$  that has polynomial stretch. Therefore, any assumption that implies such a PRG suffices for our application, and we chose the assumption that E is hard for exponential size nondeterministic circuits, for the ease of stating it. Furthermore, it is sufficient for us that  $G$  fools *uniform* AM protocols, and we don't need to fool *nonuniform* nondeterministic circuits. There is a line of work on constructing PRGs against uniform classed under uniform assumption [26, 46, 23, 39], but unfortunately, the relevant results only give hitting set generators, and using these we can only get incompressible function with  $\epsilon = 1 - n^{-O(t)}$ .



depending on the value of the coin. Merlin is supposed to guess Arthur's coin. Note that if  $h_s$  is not an  $(\ell, 2\epsilon)$ -PRG, then the two distributions  $C(U_n)$  and  $C(h_s(U_{2\ell}))$  are not  $2\epsilon$ -close and Merlin can indeed guess Arthur's coin with probability  $\frac{1}{2} + \epsilon$ . If  $h_s$  is an  $(\ell, \epsilon)$ -PRG, then the distributions are  $\epsilon$ -close and Merlin cannot distinguish with probability larger than  $\frac{1}{2} + \epsilon/2$ .

### 2.3 The power and limitations of nondeterministic reductions

The precise definitions of nondeterministic reductions and formal restatement of Theorem 1.15 appears in the full version [1]. Below, we try to intuitively explain what makes nondeterministic reductions more powerful than deterministic reductions, and why this additional power is more helpful when constructing nonboolean functions, and less helpful when constructing boolean functions.

Recall that we observed that nondeterministic reductions can be used to achieve negligible error  $\epsilon = n^{-\omega(1)}$  when constructing incomputable functions  $f : \{0, 1\}^n \rightarrow \{0, 1\}^m$  for large  $m$ , and we want to show that they cannot achieve this for  $m = 1$ . A powerful tool used by several nondeterministic reductions is *approximate counting*.

► **Theorem 2.1** (approximate counting [43, 42, 27]). *For every sufficiently large  $n$ , and every  $\epsilon' > 0$  there is a size  $\text{poly}(n/\epsilon')$  randomized NP-circuit that, given oracle access to a function  $C : \{0, 1\}^n \rightarrow \{0, 1\}$ , outputs with probability  $1 - 2^{-n}$  an integer  $p$  which  $\epsilon'$ -approximates the value  $q = |\{x : C(x) = 1\}|$  in the sense that  $(1 - \epsilon) \cdot p \leq q \leq (1 + \epsilon) \cdot p$ .*

We want the oracle circuit above to have size  $\text{poly}(n)$ , and so we can only afford  $\epsilon' = n^{-c}$ . Suppose that we are using approximate counting with this  $\epsilon'$  on some function  $C : \{0, 1\}^n \rightarrow \{0, 1\}$ , to try and distinguish the case that  $q = |\{x : C(x) = 1\}|/2^{-n}$  satisfies  $q \leq 2^{-m}$  from the case that  $q \geq 2^{-m} + \epsilon$ , for negligible  $\epsilon = n^{-\omega(1)}$ . Note that an  $n^{-c}$ -approximation can indeed perform this task distinguish if  $m \geq \log(1/\epsilon)$ , but it cannot distinguish if  $m = 1$ .

The reductions that we describe in the proofs of Theorems 1.16 and 1.21 construct functions with  $m$  bit outputs, and critically rely on this property. We now observe that in order to be useful for constructing functions with output length  $m$ , reductions must be able to distinguish the two cases above.

Let us focus on the task of constructing incomputable functions  $f : \{0, 1\}^n \rightarrow \{0, 1\}^m$ . Such reductions receive oracle access to a circuit  $C : \{0, 1\}^n \rightarrow \{0, 1\}^m$ , and if  $C$  computes  $f$  too well on average, the reduction needs to contradict the hardness assumption. Loosely speaking, we observe that the reduction must be able to distinguish the case that it is given a *useful* circuit  $C$ , namely one such that  $\Pr_{x \leftarrow U_n}[C(x) = f(x)] \geq 2^{-m} + \epsilon$  (on which the reduction must succeed) from the case that it is given a *useless* circuit  $C'$ , which ignores its input, and outputs a random value, so that  $\Pr_{x \leftarrow U_n}[C'(x) = f(x)] = 2^{-m}$  (and as this circuit is useless, the reduction receives no information on  $f$ , and cannot succeed).

This explains why approximate counting is in some sense *necessary* for reductions that want to achieve negligible error. In the formal proof, we use an argument similar to that of Furst, Saxe and Sipser [14], to show that even reductions that are  $\Sigma_i$ -circuits, cannot approximately count with the precision needed for distinguishing the cases above if  $m = 1$ . This is shown by relating the quality of such reductions to the quality of  $\text{AC}^0$ -circuits that need to perform some task (for which there are known lower bounds). This relationship uses ideas from the previous lower bounds of Shaltiel and Viola [40].



## 2.4 Constructing relative error extractors for recognizable distributions

By lemma 1.23 it is sufficient to construct relative-error extractors for recognizable distributions in order to obtain non-boolean incompressible functions with negligible error. We now explain how to construct such extractors and prove Theorem 1.24. We use tools and techniques from Trevisan and Vadhan [47], together with some key ideas that allow us to get relative error. The full proof appears in the full version [1].

It is complicated to explain the precise setting, and instead we attempt to explain what enables us to obtain relative-error. For this purpose, let us restrict our attention to the problem of constructing an  $\epsilon$ -incomputable function  $g : \{0, 1\}^n \rightarrow \{0, 1\}^m$  for  $\epsilon = n^{-c} \cdot 2^{-m}$ , which means that the function cannot be computed with probability larger than  $(1+n^{-c}) \cdot 2^{-m}$  on a random input.

We will start from a function that is already very hard on average, say  $f : \{0, 1\}^n \rightarrow \{0, 1\}^{n'}$  that is  $\epsilon$ -incomputable for  $\epsilon = 2^{-n'/3}$  (and we indeed have such a function by Theorem 1.13 for  $n' = \Omega(n)$ ). We want to reduce the output length of  $f$  from  $n'$  to  $m \approx \log(1/\epsilon)$  while preserving  $\epsilon$ . This will make  $\epsilon$  small compared to  $2^{-m}$ .

A standard way to reduce the output length while preserving security is the Goldreich-Levin theorem [17] or more generally, concatenating with a “good” inner code. More precisely, it is standard to define  $g(x, i) = EC(f(x))_i$  for some error correcting code  $EC : \{0, 1\}^{n'} \rightarrow (\{0, 1\}^m)^t$  that has sufficiently efficient list-decoding. Typically, the inner code that we use is binary (that is  $m = 1$ ). However, we want to choose codes with large alphabet that have extremely strong list decodability. One way to get such behavior is to use “extractor codes” (defined by Ta-Shma and Zuckerman [45]). More precisely, to set  $g(x, i) = T(f(x), i)$  where  $T : \{0, 1\}^{n'} \times [t] \rightarrow \{0, 1\}^m$  is a “seeded extractor”. This guarantees that for every event  $A \subseteq \{0, 1\}^m$ , there aren’t “too many”  $x$ ’s for which  $T(x, \cdot)$  lands in  $A$  with “too large probability” (this is the kind of “combinatorial list-decoding” guarantee that we are interested in). It turns out that for our application we need to replace “seeded extractors” with “2-source extractors”. A useful property of 2-source extractors is that they can achieve error  $\ll 2^{-m}$ . In particular, if applied with error  $\epsilon \ll 2^{-m}$ , such extractors can be thought of as achieving “relative error” - the probability of every output string is between  $2^{-m} - \epsilon = (1 - \epsilon \cdot 2^m) \cdot 2^{-m}$  and  $2^{-m} + \epsilon = (1 + \epsilon \cdot 2^m) \cdot 2^{-m}$ . This can be seen as a relative approximation with error  $\epsilon' = \epsilon \cdot 2^m$ .

We observe that such extractors can be used as “inner codes” in the approach of [47] (which can be viewed as a more specialized concatenation of codes). Precise details appear in the formal proof.

As in the case of Goldreich-Levin, these “codes” need to have efficient “list-decoding procedures”. In this setup “efficient” means: a list decoding procedure implementable by a polynomial size NP-circuit. In order to obtain such a list decoding procedure (for very small  $\epsilon$ ) we critically use that approximate counting can indeed distinguish  $2^{-m}$  from  $2^{-m} + \epsilon$  for negligible  $\epsilon$  using a noticeable approximation precision  $\epsilon' = n^{-c}$ , as explained in Section 2.3.

## 2.5 Relative error extractors for samplable distributions

We now explain how to construct relative error extractors for samplable distributions and prove Theorem 1.21. In this high level overview, let us restrict our attention to samplable distributions that are flat, that is uniform over some subset  $S \subseteq \{0, 1\}^n$ . Let  $X$  be such a distribution, and let  $C : \{0, 1\}^t \rightarrow \{0, 1\}^n$  be a circuit that samples  $X$  (that is  $X = C(U_t)$ ). It immediately follows that  $X$  is recognizable by the NP-circuit that given  $x$  accepts iff there exists  $y \in \{0, 1\}^t$  such that  $C(y) = x$ . This means that it suffices to construct a relative-error

extractor for distributions samplable by NP-circuits. This follows from Theorem 1.24 just the same, if in the assumption we assume hardness for  $\Sigma_4$ -circuits, instead of  $\Sigma_3$ -circuits. This follows by observing that the proof of Theorem 1.24 relativizes. The argument sketched above gives an extractor for flat samplable distributions. In order to extend this to distributions that are not flat, we generalize the notion of recognizable distributions to non-flat distributions and then Theorem 1.21 follows from the (generalized version) of Theorem 1.24.

**Acknowledgement.** We thank Yuval Ishai for helpful discussions. The first author was supported by ERC starting grant 639813 ERC-CLC, ISF grant 1155/11, Israel Ministry of Science and Technology (grant 3-9094), GIF grant 1152/2011, and the Check Point Institute for Information Security. The second author was supported by ERC starting grant 279559. The third author was supported by BSF grant 2010120, ISF grant 864/11, and ERC starting grant 279559. The work of the fourth author was done while visiting the first author in Tel-Aviv University.

---

## References

- 1 Benny Applebaum, Sergei Artemenko, Ronen Shaltiel, and Guang Yang. Incompressible functions, relative-error extractors, and the power of nondeterministic reductions. *Electronic Colloquium on Computational Complexity (ECCC)*, 15(51), 2015.
- 2 Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. From secrecy to soundness: Efficient verification via secure computation. In *ICALP*, pages 152–163, 2010.
- 3 Benny Applebaum, Yuval Ishai, Eyal Kushilevitz, and Brent Waters. Encoding functions with constant online rate or how to compress garbled circuits keys. In *CRYPTO*, pages 166–184, 2013.
- 4 S. Artemenko and R. Shaltiel. Lower bounds on the query complexity of non-uniform and adaptive reductions showing hardness amplification. *Computational Complexity*, 23(1):43–83, 2014.
- 5 Sergei Artemenko and Ronen Shaltiel. Pseudorandom generators with optimal seed length for non-boolean poly-size circuits. In *Symposium on Theory of Computing, STOC*, pages 99–108, 2014.
- 6 L. Babai, L. Fortnow, N. Nisan, and A. Wigderson. Bpp has subexponential time simulations unless exponential time has publishable proofs. *Computational Complexity*, 3:307–318, 1993.
- 7 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.
- 8 B. Barak, S. J. Ong, and S. P. Vadhan. Derandomization in cryptography. *SIAM J. Comput.*, 37(2):380–400, 2007.
- 9 Kai-Min Chung, Yael Kalai, and Salil Vadhan. Improved delegation of computation using fully homomorphic encryption. In *CRYPTO*, pages 483–501, 2010.
- 10 Francesco Davì, Stefan Dziembowski, and Daniele Venturi. Leakage-resilient storage. In *Security and Cryptography for Networks, 7th International Conference, SCN 2010*, pages 121–137, 2010.
- 11 Andrew Drucker. Nondeterministic direct product reductions and the success probability of SAT solvers. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS*, pages 736–745, 2013.
- 12 B. Dubrov and Y. Ishai. On the randomness complexity of efficient sampling. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, pages 711–720, 2006.
- 13 Uriel Feige and Carsten Lund. On the hardness of computing the permanent of random matrices. *Computational Complexity*, 6(2):101–132, 1997.

- 14 Merrick L. Furst, James B. Saxe, and Michael Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical Systems Theory*, 17(1):13–27, 1984.
- 15 Rosario Gennaro, Craig Gentry, and Bryan Parno. Non-interactive verifiable computing: Outsourcing computation to untrusted workers. In *CRYPTO*, pages 465–482, 2010.
- 16 O. Goldreich and A. Wigderson. Derandomization that is rarely wrong from short advice that is typically good. In *APPROX-RANDOM*, pages 209–223, 2002.
- 17 Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing*, pages 25–32, 1989.
- 18 Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that yield nothing but their validity for all languages in NP have zero-knowledge proof systems. *J. ACM*, 38(3):691–729, 1991.
- 19 Shafi Goldwasser and Michael Sipser. Private coins versus public coins in interactive proof systems. In *Proceedings of the 18th Annual ACM Symposium on Theory of Computing*, pages 59–68, 1986.
- 20 D. Gutfreund and G. Rothblum. The complexity of local list decoding. In *12th Intl. Workshop on Randomization and Computation (RANDOM)*, 2008.
- 21 D. Gutfreund, R. Shaltiel, and A. Ta-Shma. If np languages are hard on the worst-case, then it is easy to find their hard instances. *Computational Complexity*, 16(4):412–441, 2007.
- 22 D. Gutfreund and A. Ta-Shma. Worst-case to average-case reductions revisited. In *APPROX-RANDOM*, pages 569–583, 2007.
- 23 Dan Gutfreund, Ronen Shaltiel, and Amnon Ta-Shma. Uniform hardness versus randomness tradeoffs for arthur-merlin games. *Computational Complexity*, 12(3-4):85–130, 2003.
- 24 Danny Harnik and Moni Naor. On the compressibility of  $\mathcal{NP}$  instances and cryptographic applications. *SIAM J. Comput.*, 39(5):1667–1713, 2010.
- 25 R. Impagliazzo and A. Wigderson.  $P = BPP$  if  $E$  requires exponential circuits: Derandomizing the XOR lemma. In *STOC*, pages 220–229, 1997.
- 26 R. Impagliazzo and A. Wigderson. Randomness vs. time: De-randomization under a uniform assumption. In *39th Annual Symposium on Foundations of Computer Science*. IEEE, 1998.
- 27 M. Jerrum, L. G. Valiant, and V. V. Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theor. Comput. Sci.*, 43:169–188, 1986.
- 28 Yael Tauman Kalai, Ran Raz, and Ron D. Rothblum. How to delegate computations: the power of no-signaling proofs. In *STOC*, 2014.
- 29 A. Klivans and D. van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses. *SIAM J. Comput.*, 31(5):1501–1526, 2002.
- 30 R. Lipton. New directions in testing. In *Proceedings of DIMACS Workshop on Distributed Computing and Cryptography*, volume 2, pages 191–202. ACM/AMS, 1991.
- 31 C.-J. Lu, S.-C. Tsai, and H.-L. Wu. On the complexity of hardness amplification. *IEEE Transactions on Information Theory*, 54(10):4575–4586, 2008.
- 32 Chi-Jen Lu, Shi-Chun Tsai, and Hsin-Lung Wu. Impossibility results on weakly black-box hardness amplification. In *FCT*, pages 400–411, 2007.
- 33 P. Bro Miltersen and N. V. Vinodchandran. Derandomizing arthur-merlin games using hitting sets. *Computational Complexity*, 14(3):256–279, 2005.
- 34 N. Nisan and A. Wigderson. Hardness vs. randomness. *JCSS: Journal of Computer and System Sciences*, 49, 1994.
- 35 R. Shaltiel. Recent developments in explicit constructions of extractors. *Bulletin of the EATCS*, 77:67–95, 2002.
- 36 R. Shaltiel. An introduction to randomness extractors. In *Automata, Languages and Programming - 38th International Colloquium*, pages 21–41, 2011.

- 37 R. Shaltiel and C. Umans. Simple extractors for all min-entropies and a new pseudorandom generator. *J. ACM*, 52(2):172–216, 2005.
- 38 R. Shaltiel and C. Umans. Pseudorandomness for approximate counting and sampling. *Computational Complexity*, 15(4):298–341, 2006.
- 39 R. Shaltiel and C. Umans. Low-end uniform hardness versus randomness tradeoffs for am. *SIAM J. Comput.*, 39(3):1006–1037, 2009.
- 40 R. Shaltiel and E. Viola. Hardness amplification proofs require majority. *SIAM J. Comput.*, 39(7):3122–3154, 2010.
- 41 Ronen Shaltiel. Weak derandomization of weak algorithms: Explicit versions of yao’s lemma. *Computational Complexity*, 20(1):87–143, 2011.
- 42 M. Sipser. A complexity theoretic approach to randomness. In *STOC*, pages 330–335, 1983.
- 43 L. J. Stockmeyer. The complexity of approximate counting. In *STOC*, pages 118–126, 1983.
- 44 M. Sudan, L. Trevisan, and S. P. Vadhan. Pseudorandom generators without the xor lemma. *J. Comput. Syst. Sci.*, 62(2):236–266, 2001.
- 45 A. Ta-Shma and D. Zuckerman. Extractor codes. In *STOC*, 2001.
- 46 L. Trevisan and S. Vadhan. Pseudorandomness and average-case complexity via uniform reductions. *Computational Complexity*, 16(4):331–364, 2007.
- 47 L. Trevisan and S. P. Vadhan. Extracting randomness from samplable distributions. In *41st Annual Symposium on Foundations of Computer Science*, pages 32–42, 2000.
- 48 E. Viola. The complexity of constructing pseudorandom generators from hard functions. *Computational Complexity*, 13(3-4):147–188, 2005.

# On Randomness Extraction in $\mathcal{AC}^0$

Oded Goldreich<sup>1</sup>, Emanuele Viola<sup>2</sup>, and Avi Wigderson<sup>3</sup>

- 1 Department of Computer Science, Weizmann Institute of Science  
Rehovot, Israel  
oded.goldreich@weizmann.ac.il
- 2 College of Computer and Information Science, Northeastern University  
Boston, MA 02115, USA  
viola@ccs.neu.edu
- 3 School of Mathematics, Institute for Advanced Study  
Princeton, NJ 08540, USA  
avi@ias.edu

---

## Abstract

We consider randomness extraction by  $\mathcal{AC}^0$  circuits. The main parameter,  $n$ , is the length of the source, and all other parameters are functions of it. The additional extraction parameters are the min-entropy bound  $k = k(n)$ , the seed length  $r = r(n)$ , the output length  $m = m(n)$ , and the (output) deviation bound  $\epsilon = \epsilon(n)$ .

For  $k \leq n/\log^{\omega(1)} n$ , we show that  $\mathcal{AC}^0$ -extraction is possible if and only if  $\frac{m}{r} \leq 1 + \text{poly}(\log n) \cdot \frac{k}{n}$ ; that is, the extraction rate  $m/r$  exceeds the trivial rate (of one) by an additive amount that is proportional to the min-entropy rate  $k/n$ . In particular, non-trivial  $\mathcal{AC}^0$ -extraction (i.e.,  $m \geq r + 1$ ) is possible if and only if  $k \cdot r > n/\text{poly}(\log n)$ . For  $k \geq n/\log^{O(1)} n$ , we show that  $\mathcal{AC}^0$ -extraction of  $r + \Omega(r)$  bits is possible when  $r = O(\log n)$ , but leave open the question of whether more bits can be extracted in this case.

The impossibility result is for constant  $\epsilon$ , and the possibility result supports  $\epsilon = 1/\text{poly}(n)$ . The impossibility result is for (possibly) non-uniform  $\mathcal{AC}^0$ , whereas the possibility result holds for uniform  $\mathcal{AC}^0$ . All our impossibility results hold even for the model of bit-fixing sources, where  $k$  coincides with the number of non-fixed (i.e., random) bits.

We also consider deterministic  $\mathcal{AC}^0$  extraction from various classes of restricted sources. In particular, for any constant  $\delta > 0$ , we give explicit  $\mathcal{AC}^0$  extractors for  $\text{poly}(1/\delta)$  independent sources that are each of min-entropy rate  $\delta$ ; and four sources suffice for  $\delta = 0.99$ . Also, we give non-explicit  $\mathcal{AC}^0$  extractors for bit-fixing sources of entropy rate  $1/\text{poly}(\log n)$  (i.e., having  $n/\text{poly}(\log n)$  unfixed bits). This shows that the known analysis of the “restriction method” (for making a circuit constant by fixing as few variables as possible) is tight for  $\mathcal{AC}^0$  even if the restriction is picked deterministically depending on the circuit.

**1998 ACM Subject Classification** F.1.1 Models of Computation

**Keywords and phrases**  $\mathcal{AC}^0$ , randomness extractors, general min-entropy sources, block sources, bit-fixing sources, multiple-source extraction

**Digital Object Identifier** 10.4230/LIPIcs.CCC.2015.601

## 1 Introduction

Randomness extractors, hereafter referred to as extractors, are procedures that transform sources of “weak randomness” into sources of almost perfect randomness. The feasibility of such a transformation depends on the specific notion of “weak randomness”, and in most cases the transformation must be provided with a short (perfectly) random seed. Indeed, this



© Oded Goldreich, Emanuele Viola, and Avi Wigderson;  
licensed under Creative Commons License CC-BY

30th Conference on Computational Complexity (CCC'15).

Editor: David Zuckerman; pp. 601–668



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

fundamental problem has several versions, and each of them comes with a set of parameters. (See Shaltiel’s survey [53] for a wide perspective as well as for a snapshot of the state of the art a decade ago.)

The most popular and general notion of weak randomness is parameterized by a probability bound, denoted  $2^{-k}$ , such that no outcome may appear with probability that exceeds it. In such a case,  $k$  is called the min-entropy of the source. Additional parameters of the extraction problem include the length of the source, denoted  $n$ , the length of the seed, denoted  $r$ , the length of the (extracted) output, denoted  $m$ , and an upper bound on its deviation (from perfect randomness), denoted  $\epsilon$ . In fact,  $n$  is viewed as the main parameter, and all other parameters are stated as functions of  $n$ . A function  $E : \{0, 1\}^n \times \{0, 1\}^r \rightarrow \{0, 1\}^m$  is called a  $(k, \epsilon)$ -extractor if for every  $X$  of min-entropy  $k$  it holds that  $E(X, U_r)$  is  $\epsilon$ -close to  $U_m$ , where  $U_\ell$  denotes the uniform distribution over  $\{0, 1\}^\ell$ . It is called a **strong**  $(k, \epsilon)$ -extractor if for such  $X$ ’s it holds that  $E(X, U_r) \circ U_r$  is  $\epsilon$ -close to  $U_{m+r}$ . (Note that if  $E$  is a strong  $(k, \epsilon)$ -extractor, then  $E'(x, u) = E(x, u) \circ u$  is an  $(k, \epsilon)$ -extractor.)

When ignoring computational issues, the exact trade-off between the various extraction parameters is known, but much research has been devoted to obtaining explicit constructions that approach the optimal bounds. Traditionally, an extractor is called explicit if it can be computed efficiently (i.e., in polynomial-time) or alternatively if Boolean circuits computing it can be constructed in  $\text{poly}(n)$ -time. It is known that some constructions are even more explicit than that; for example, the popular constructions of universal hashing functions [15] (known as the “mother of all extractors”) are computable by highly uniform  $\mathcal{AC}^0[2]$  circuits (i.e., constant-depth circuits of polynomial-size with parity gates). The same holds for Trevisan’s celebrated extractor [54]. *Can one get any lower* (indeed to  $\mathcal{AC}^0$ )? This is the question we study here.<sup>1</sup>

## 1.1 The most relevant prior work

Our starting point is the following negative result by Viola [57].

► **Theorem 1.1** (severe limitations on extraction in  $\mathcal{AC}^0$  [57, Thm. 6.4]). *If a  $(k, 0.999)$ -extractor  $E : \{0, 1\}^n \times \{0, 1\}^{0.999m} \rightarrow \{0, 1\}^m$  is computable by a circuit  $C$  (with negations and unbounded fan-in and and or gates), then  $\text{size}(C) \geq \exp(\Omega(n/k)^{1/(\text{depth}(C)-1)})$ . In particular, if  $E$  can be computed by a family of  $\mathcal{AC}^0$  circuits, then there exists a positive polynomial  $p$  such that  $k(n) \geq n/p(\log n)$  for all sufficiently large  $n$ .*

This result rules out  $\mathcal{AC}^0$ -extractors that either extract from entropy  $k = n/\log^{\omega(1)} n$  or use a seed length  $r$  that is sublinear in the output length. However, the result leaves open the possibility that (non-trivial)  $\mathcal{AC}^0$ -extractors exist for other settings of parameters. Indeed, when making only the non-triviality requirement (i.e.,  $m = r + 1$ ), two such extractors existed in the literature. First,  $\mathcal{AC}^0$  circuits can extract one bit from a source of logarithmic min-entropy when using a very long seed; specifically, when  $r = n = m - 1$ . (This can be done by sampling input-output pairs of the inner product function [34], cf. [6, 58].) Second, non-trivial  $\mathcal{AC}^0$ -extractors exist for min-entropy  $k \geq n/\text{poly} \log n$  (using the “sample-then-

<sup>1</sup> In fact, one may even go lower and ask whether randomness extraction is possible in  $\mathcal{NC}^0$ . Actually, such extractors were presented in [5, Sec. 5.3] for the case of  $m = n$  and  $r = \Theta(n - k)$ . We note that these parameters are inferior to the parameters in Theorem 1.7, but they are the best possible for  $\mathcal{NC}^0$  (since if an extractor of locality  $d$  extracts  $m = r + 1$  bits such that  $d \cdot m \leq n - k$ , then the entropy of the output bits must come from the seed because  $n - k$  bits of the source may be fixed).



extract” paradigm developed by Nisan and Zuckerman [46] and refined by Vadhan [55]). In fact, this  $\mathcal{AC}^0$ -extractor can extract logarithmically many bits.

► **Theorem 1.2** (following [55, Thm. 7.4]). *For every  $k(n) = n/\text{poly}(\log n)$  and  $\epsilon(n) = 1/\text{poly}(n)$ , there exist (non-explicit)  $\mathcal{AC}^0$  circuits that compute a strong  $(k(n), \epsilon(n))$ -extractor  $E : \{0, 1\}^n \times \{0, 1\}^{O(\log n)} \rightarrow \{0, 1\}^{\Theta(\log n)}$ . Furthermore, the circuits have depth  $4 + \left\lceil \frac{\log(n/k(n))}{\log \log n} \right\rceil$ .*

The proof of Theorem 1.2 follows the proof of [55, Thm. 7.4], which combines an adequate sampler with an adequate extractor using the (sample-then-extract) composition theorem of [55, Thm. 6.3]. We note that both the sampler and the extractor used in the original proof of [55, Thm. 7.4] are non-explicit; furthermore, the (optimal) extractor used there is probably not computable by constant-depth circuits of  $\text{poly}(n)$ -size (let alone explicit ones). Instead, we shall use the (non-optimal) explicit extractor of [29, Sec. 5], which is computable by (uniform) constant-depth circuits of size  $\text{poly}(n)$ . The resulting  $\mathcal{AC}^0$ -extractor inherits the non-explicitness of the sampler used in the proof of [55, Thm. 7.4]. Jumping ahead, we mention that we provide an explicit version of Theorem 1.2 (see Theorem 3.1) by using a new explicit sampler (see Theorem 3.2).

Hence, while a non-explicit  $\mathcal{AC}^0$ -extractor for  $k \geq n/\text{poly}(\log n)$  is implicit in prior work, the explicit version (as stated in Theorem 3.1) relies on our new sampler (i.e., Theorem 3.2). In any case, the foregoing results do not refer to the general trade-offs between the parameters  $k$ ,  $r$  and  $m$  that allow extraction to be performed in  $\mathcal{AC}^0$ .

## 1.2 Our main results

We study the following general question.

**Parameters enabling extraction in  $\mathcal{AC}^0$ :** *For which values of  $k$ ,  $r$  and  $m$  are  $(k, n^{-3})$ -extractors  $E : \{0, 1\}^n \times \{0, 1\}^{r(n)} \rightarrow \{0, 1\}^{m(n)}$  computable in  $\mathcal{AC}^0$ ?*

Recall that, for logarithmic seed length (i.e.,  $r(n) = O(\log n)$ ), a min-entropy bound of  $k(n) > n/\text{poly}(\log n)$  is a necessary condition (even for  $m(n) = r(n) + \Omega(\log n)$  (see Theorem 1.1)), whereas  $m(n) = r(n) + O(\log n)$  is achievable (by Theorem 1.2). These results mark the boundaries (between impossible and possible) as a function of  $k$  when  $r(n) = O(\log n)$  and  $m(n) = r(n) + \Theta(\log n)$ . Indeed, this boundary refers to a line  $(k, r = O(\log n), m = r + \Theta(\log n))$  in the three dimensional space  $(k, r, m) \in [n]^3$ . Our work is aimed at mapping the entire space, and it achieves this goal for  $k < n/\log^{\omega(1)} n$ . *In this setting, we show that  $\mathcal{AC}^0$ -extraction is possible if and only if  $\frac{m}{r} \leq 1 + \text{poly}(\log n) \cdot \frac{k}{n}$ ; that is, the extraction rate  $m/r$  exceeds 1 by an additive amount that is proportional to the min-entropy rate  $k/n$ .* The region of  $k \geq n/\text{poly}(\log n)$  remains partially unmapped (as indicated in Problem 1.6 below).

In general, our impossibility results are for constant  $\epsilon$ , and the possibility results support  $\epsilon = 1/\text{poly}(n)$ . The impossibility results are for (possibly non-uniform)  $\mathcal{AC}^0$ , whereas all but one of our possibility results hold for uniform  $\mathcal{AC}^0$  (assuming that the relevant functions (i.e.,  $k, r, m$  and  $\epsilon$ ) are  $\text{poly}(n)$ -time computable). *All impossibility results hold even for the model of bit-fixing sources, where  $k$  coincides with the number of non-fixed (i.e., random) bits.*

When we write  $k(n) < n/\text{poly}(\log n)$  we mean that, for every positive polynomial  $p$  and all sufficiently large  $n$ , it holds that  $k(n) < n/p(\log n)$ . When we write  $k(n) \geq n/\text{poly}(\log n)$  we mean that there exists a polynomial  $p$  such that, for all sufficiently large  $n$ , it holds that  $k(n) \geq n/p(\log n)$ .



With these preliminaries in place, we turn to describe our main results. In the case of strong extraction we obtain a very clear dichotomy.

► **Theorem 1.3** (strong extraction in  $\mathcal{AC}^0$ ).

- impossibility: *Strong extraction, even of a single bit, is impossible in  $\mathcal{AC}^0$  for  $k(n) < n/\text{poly}(\log n)$ , regardless of the length of the seed.*
- possibility: *Strong extraction of  $m_0 = \Omega(\log n)$  bits is possible in uniform  $\mathcal{AC}^0$  for any  $k(n) \geq n/\text{poly}(\log n)$ , using a seed of length  $O(\log n)$ . Furthermore, in this case, for every  $t < k/2m_0$ , strong extraction of  $t \cdot m_0$  bits is possible using a seed of length  $O(t \cdot \log n)$ .*

[The impossibility result follows by Theorem 5.4. The possibility result follows by Part 2 of Corollary 6.4, which in turn is based on Theorem 3.1 (combined with Theorem 6.3 for the furthermore part).]

We comment that, for  $k(n) \geq n/\text{poly}(\log n)$ , one can extract poly-logarithmically many bits in  $\mathcal{AC}^0$  using a seed of logarithmic length but at an error rate of  $1/\text{poly}(\log n)$ ; this can be done by using Trevisan's extractors [54] (instead of the extractor of [29, Sec. 5]). We now turn to ordinary (i.e., non-strong) extraction, starting with the minimal case of non-trivial extraction.

► **Theorem 1.4** (ordinary extraction in  $\mathcal{AC}^0$ , the case of  $m = r + 1$ ).

- impossibility: *Extraction of  $r(n)+1$  bits is impossible in  $\mathcal{AC}^0$  for  $r(n) \cdot k(n) < n/\text{poly}(\log n)$ .*
- possibility: *There exists a constant  $c > 2$  such that, for every  $k(n) \geq c \cdot \log(n/\epsilon)$  and every  $r(n) \geq (n \cdot \log^3 n)/k(n)$ , extraction of  $r(n) + \min(\text{poly}(\log n), k(n)/2)$  bits is possible in uniform  $\mathcal{AC}^0$ .*

[The impossibility result follows by Part 1 of Theorem 5.5, whereas the possibility result follows by Corollary 6.2.]

► **Theorem 1.5** (ordinary extraction in  $\mathcal{AC}^0$ , the case of  $m = r + \Theta(r)$ ).

- impossibility: *Extraction of  $r(n) + \Omega(r(n))$  bits is impossible in  $\mathcal{AC}^0$  for  $k(n) < n/\text{poly}(\log n)$ , regardless of  $r$ .*
- possibility: *There exists a constant  $c > 0$  such that, for every  $k(n) \geq n/\text{poly}(\log n)$  and every  $r(n) \in [\Omega(\log n), k(n)/c]$ , extraction of  $(1 + c) \cdot r(n)$  bits is possible in uniform  $\mathcal{AC}^0$  using a seed of length  $r(n)$ .*

[The impossibility result follows by Part 2 of Theorem 5.5, whereas the possibility result (of Theorem 1.5) follows by the possibility result of Theorem 1.3.]

Note that the impossibility result of Theorem 5.5 establishes the same bound as Viola's [57, Thm. 6.4] (see Theorem 1.1), but does so for bit-fixing sources.

Let us restate the message of Theorem 1.5: It says that extracting  $m(n) = r(n) + \Theta(r(n))$  bits is impossible if  $k(n) < n/\text{poly}(\log n)$  (regardless of the size of  $r$ ), whereas if  $k(n) \geq n/\text{poly}(\log n)$  then using a seed of length  $r(n) = \Omega(\log n)$  we can extract  $r(n) + \Omega(r(n))$  bits. However, it is not clear whether we cannot extract significantly more bits in the latter case.

► **Open Problem 1.6** (extracting more bits at rate of at least  $1/\text{poly}(\log n)$ ). *Can one extract more than  $\text{poly}(\log n) \cdot r(n)$  bits in  $\mathcal{AC}^0$  using a seed of length  $r(n) = \Omega(\log n)$ , when  $k(n) > n/\text{poly}(\log n)$ ? In particular, can one extract more than  $\text{poly}(\log n)$  bits using a seed of logarithmic length? For starters, what about the special case of constant min-entropy rate, that is,  $k(n) = \Omega(n)$ ?*

We conjecture that the answer is negative and provide some evidence for this conjecture in Section 4.2. Recall that for  $k(n) \geq n/\text{poly}(\log n)$ , we can extract poly-logarithmically many bits using a seed of logarithmic seed but at an error rate of  $1/\text{poly}(\log n)$ ; see Part 2

of Corollary 3.6. Indeed, a minor open problem regarding this case is to reduce the error rate to  $1/\text{poly}(n)$ .

Theorems 1.4 and 1.5 can be interpolated, and they indeed follow as special cases of the following generalization (which is our main result).

- **Theorem 1.7** (ordinary extraction in  $\mathcal{AC}^0$ , the general case of  $m = r + m'$ ).
- **impossibility:** For any  $m'(n) \geq 1$ , extraction of  $r(n) + m'(n)$  bits is impossible in  $\mathcal{AC}^0$  if  $k(n) < \frac{m'(n)}{r(n)+m'(n)} \cdot \frac{n}{\text{poly}(\log n)}$  (equiv., if  $(r(n) + m'(n)) \cdot k(n) < m'(n) \cdot n/\text{poly}(\log n)$ ).
  - **possibility:** There exists a constant  $c > 1$  such that for every  $k(n)$ ,  $m'(n)$ , and  $r(n)$  such that  $k(n) \geq c \cdot (m'(n) + \log(n/\epsilon))$ , extraction of  $r(n) + m'(n)$  bits is possible in uniform  $\mathcal{AC}^0$  in each of the following two cases.
    1. For  $r(n) \cdot k(n) \geq \lceil m'(n)/\text{poly}(\log n) \rceil \cdot O(n \log^2 n)$  and  $k(n) > \text{poly}(\log n)$ ;
    2. For  $r(n) = n$ . Furthermore, for  $r(n) \cdot k(n) \geq n/\text{poly}(\log n)$ , we have  $m'(n) = \Omega(k(n)) - \text{poly}(\log n)$ .

[The impossibility result follows by Theorem 5.5, whereas the possibility result follows by Corollary 6.4.]

The result of Theorem 1.7 is almost tight for  $k(n) < n/\text{poly}(\log n)$ : Extraction of  $r(n) + m'(n)$  bits is impossible in  $\mathcal{AC}^0$  if  $r(n) \cdot k(n) + m'(n) \cdot k(n) < m'(n) \cdot n/\text{poly}(\log n)$ , which is equivalent (in this case) to  $r(n) \cdot k(n) < m'(n) \cdot n/\text{poly}(\log n)$ , but is possible if  $r(n) \cdot k(n) \geq m'(n) \cdot n/\text{poly}(\log n)$  (provided that  $\text{poly}(\log n) < m'(n) \leq k(n) - O(\log n)$ ). Hence, what we really do not know refers to the range of  $k(n) \geq n/\text{poly}(\log n)$ ; that is, to Problem 1.6.

A different perspective on Theorem 1.7 is obtained by considering the relation between  $k/n$  and  $m'/r$ .

Theorem 1.7 asserts that for  $m'/r \in [0, \Theta(1)]$ , extraction in  $\mathcal{AC}^0$  is possible if  $k/n \geq f(n) \cdot m'/r$  for some  $f(n) = 1/\text{poly}(\log n)$  and impossible if  $k/n < f'(n) \cdot m'/r$  for every  $f'(n) = 1/\text{poly}(\log n)$ . Recall that Theorems 1.1 and 1.2 only refer to the case of  $m'/r = \Theta(1)$ , whereas Theorem 1.7 covers all  $m'/r \in [0, \Theta(1)]$ . Problem 1.6 refers to  $m'/r = \omega(1)$  (or actually to  $m'/r = (\log n)^{\omega(1)}$ ).

We highlight the fact that non-trivial extraction (i.e.,  $m'(n) = 1$ ) is possible in  $\mathcal{AC}^0$  if and only if  $k \cdot r > n/\text{poly}(\log n)$ . In contrast, the threshold for “significant”  $\mathcal{AC}^0$ -extraction, that is  $m' = \Omega(r)$ , is  $k > n/\text{poly}(\log n)$ . Hence, for  $r > \text{poly}(\log n)$ , there is a gap between the min-entropy bound that allows non-trivial  $\mathcal{AC}^0$ -extraction and the min-entropy required for extracting  $r + \Omega(r)$  bits in  $\mathcal{AC}^0$ .

### Extraction with respect to restricted sources

In relation to Problem 1.6, we mention that both in the model of block sources and in the model of bit-fixing sources, we can *extract more than poly-logarithmically many bits using a seed of logarithmic length*, where in both cases the min-entropy rate is at least  $1/\text{poly}(\log n)$  (and extraction is in  $\mathcal{AC}^0$ ). In fact, in both cases,  $n/\text{poly}(\log n)$  bits are extracted.

- **Theorem 1.8** (extraction in  $\mathcal{AC}^0$  for bit-fixing sources and block sources). For any  $k(n) \geq n/\text{poly}(\log n)$ , extraction of  $n/\text{poly}(\log n)$  bits using a seed of length  $O(\log n)$  is possible in uniform  $\mathcal{AC}^0$  for bit-fixing sources in which  $k(n)$  of the  $n$  bits are not fixed. Ditto for block sources with  $\Theta(k(n)/\log n)$  blocks such that each block has conditional min-entropy  $\Omega(\log n)$ .

[See Corollary 4.3 and Theorem 5.2.]

Recall that the bit-fixing model allows for deterministic extractors, which work even for lower min-entropy rates, but these extractors are not computable by  $\mathcal{AC}^0$  (which is to be

expected in light of the fact that our impossibility results hold for the bit-fixing model). Still, it is possible that whenever extraction in  $\mathcal{AC}^0$  is possible for bit-fixing sources, it is also possible via deterministic extractors. We show that this is essentially the case.

► **Theorem 1.9** (deterministic extraction in  $\mathcal{AC}^0$  for bit-fixing sources). *For any  $k(n) \geq n/\text{poly}(\log n)$ , deterministic extraction of  $n/\text{poly}(\log n)$  bits is possible in  $\mathcal{AC}^0$  for bit-fixing sources in which  $k(n)$  of the  $n$  bits are not fixed.*

[See Theorem 5.8.] Unlike all previously mentioned results, this possibility result only claims the existence of  $\mathcal{AC}^0$  circuits (but does not provide an explicit construction). We mention (see Theorem 5.18) that we can construct explicit  $\mathcal{AC}^0$  circuits that compute a deterministic disperser for this class of sources (i.e., we present a circuit that is not constant on any such source).

**A circuit complexity perspective (w.r.t random restrictions).** A (deterministic) extractor for bit-fixing sources in which  $k(n)$  of the  $n$  bits are random constitutes a circuit that is not trivialized (i.e., does not simplify to a constant) under any restriction that keeps  $k(n)$  of the variables alive.<sup>2</sup> Hence, Hastad's analysis [32] of the random restriction method [1, 24, 60], which implies that any depth  $d$  circuit of size  $s(n)$  trivializes under a random restriction that keeps  $n/O(\log^{d-1} s(n))$  variables alive, is optimal in a very strong sense: Not only that there exist  $\mathcal{AC}^0$  circuits that do not trivialize *under a random restriction* that keeps  $n/\text{poly}(\log n)$  variables alive, but these circuits are not trivialized *under any restriction* that keeps  $n/\text{poly}(\log n)$  variables alive. In other words, *a restriction that is carefully selected based on the target circuit cannot achieve significantly better parameters than a random restriction* (i.e., cannot trivialize the circuit while leaving significantly more variables alive). There are contexts in which circuit-dependent restriction yields stronger lower bounds than its randomized counterpart. These contexts include  $\mathcal{AC}^0$  circuits of nearly-linear size [16], and of threshold circuits of nearly-linear size [35].

**Deterministic extraction from several independent sources.** Another model allowing for deterministic extractors is the model of two or more independent sources each having min-entropy at least  $k(n)$

(cf., e.g., [17, 7, 8, 38]). While this model allows for deterministic extractors, which work even for min-entropy rates below  $1/\text{poly}(\log n)$ , the known extractors are not computable by  $\mathcal{AC}^0$  (which is to be expected in light of the fact that our impossibility results hold also for this model). We show that deterministic extraction in  $\mathcal{AC}^0$  is possible also in this model.

► **Theorem 1.10** (deterministic extraction in  $\mathcal{AC}^0$  for the multi-source model). *For any constant  $\delta > 0$ , there exist explicit  $\mathcal{AC}^0$ -extractors for  $\text{poly}(1/\delta)$  independent sources that are each of min-entropy rate  $\delta$ . For  $\delta = 0.99$ , four sources suffice.*

See Section 7. In the two-source model, we only obtain such extractors for rates that approach 1 (i.e.,  $\delta \geq 1 - \log^{-4} n$ ).

### 1.3 Techniques

Our results build on known results and known techniques. These are augmented by several new constructions of various pseudorandom objects including

<sup>2</sup> In fact, the same holds for dispersers, which are functions that map any such source to a non-trivial distribution.

- two alternative constructions of averaging samplers (see Sections 3.2 and 3.4, respectively);
- an  $\mathcal{AC}^0$ -extractor for bit-fixing sources extracting  $n/\text{poly}(\log n)$  bits using a logarithmically long seed (see Section 5.1);
- a deterministic  $\mathcal{AC}^0$ -extractor for bit-fixing sources extracting  $\text{poly}(\log n)$  bits (see Section 5.3);
- a deterministic two-source  $\mathcal{AC}^0$ -extractor (see Section 7.1);
- deterministic many-source  $\mathcal{AC}^0$ -extractors for lower entropy rate (see Sections 7.2 and 7.3).

These and other contributions are mentioned in Section 1.5, *where we emphasize interesting aspects that are not mentioned here*. In the current section, we focus on a few common themes that re-occur in several proofs. Similar ideas were used before, but we found the current incarnations useful and worthy of highlighting.

### Generating pseudorandom partitions

Loosely speaking, the problem is to generate a pseudorandom partition of  $[n]$  into  $m$  equal-sized sets such that each set has a strong hitting or sampling property. We wish to do this using a logarithmic amount of randomness and for  $m = n/\text{poly}(\log n)$ . Specifically, Lemma 5.3 asserts a pseudorandom partition generator for  $n/m = O(\rho^{-1} \log(1/\epsilon))^2$  such that each set of density  $\rho$  is hit by each subset of the partition with probability at least  $1 - \epsilon$ .

The idea is to use a fixed partition of  $[n]$  into  $n/m$  disjoint  $m$ -cycles and a standard hitter of sample complexity  $\sqrt{n/m}$ . Hoping that this hitter generates a set that hits each cycle at most once, we augment this set to a cover of all  $n/m$  cycles, and use the  $m$  “shifts” of this augmented set as a partition. Using a suitable implementation, the aforementioned hope does materialize with constant probability, and we augment the construction so to obtain a good partition with overwhelmingly high probability. (The proof of Lemma 5.3 presents a specific instantiation of this idea that is implementable in uniform  $\mathcal{AC}^0$ .)

Somewhat related problems arise in the proofs of Theorems 3.2 and 3.8. In these proofs, we need to construct a sampler that generates a very large set (say of size  $n^{1/3}$ ) of *distinct elements*. This is relatively easy if the elements of the sample are pairwise independent. Getting sets of size greater than  $\sqrt{n}$  requires additional ideas, which appear in the proof of Theorem 3.8.

In the latter case we rely on the fact that the number of occurrences of each element in the sample is not large, and that this number can be computed using a high quality hashing scheme. We then include each element in the final sample with probability that is proportional to the number of occurrences in the first sample, while noting that this random sieving preserves the sampling property of the first sample. (We warn that the implementation of this procedure in  $\mathcal{AC}^0$  is not straightforward.)

### Combining various pseudorandom properties

As hinted above, we may want to have a good sampler that uses samples that are uniformly distributed in the domain in a  $O(1)$ -wise independent manner. The problem is that good samplers use random walks on expander graphs, and in this case the samples are each uniformly distributed but they are not even pairwise independent.

The solution is to XOR an  $O(1)$ -wise independent sequence with the vertices visited in the random walk (see Claim 3.4). We show that the combined sampler inherits the properties of each of the original samplers. Indeed, such combinations were used before for different properties (e.g., Impagliazzo and Wigderson [36] de-randomized Yao’s XOR Lemma by

XORing the output of the “projected seed generator” of [45] with the output of a random walk generator).

#### 1.4 The perspective of error reduction

As articulated by Zuckerman [63], there is a close relation between randomness extractors that are computable in a natural complexity class such as  $\mathcal{AC}^0$  and error-reduction procedures *computable in that class*. The error-reduction procedures referred to here are confined to generating several inputs, applying the original circuit to each of these inputs, and ruling by majority. Hence, these procedures are closely related to averaging samplers (as defined implicitly in the next paragraph).

Specifically, a  $(k, 0.1)$ -extractor  $E : \{0, 1\}^n \times \{0, 1\}^r \rightarrow \{0, 1\}^m$  yields a sampler  $S : \{0, 1\}^n \rightarrow (\{0, 1\}^m)^{2^r}$  such that for every  $f : \{0, 1\}^m \rightarrow \{0, 1\}$  with probability at least  $1 - 2 \cdot 2^{-(n-k)}$  it holds that  $2^{-r} \cdot \sum_{s \in S(U_n)} f(s) = (1 \pm 0.1) \cdot \mathbf{E}[f(U_m)]$ . (Just use  $S(x) = \{E(x, \sigma) : \sigma \in \{0, 1\}^r\}$ .) The converse holds too (by using  $E(x, \sigma) = S(x)_\sigma$ ): A sampler  $S : \{0, 1\}^n \rightarrow (\{0, 1\}^m)^{2^r}$  that satisfies  $\Pr[2^{-r} \cdot \sum_{s \in S(U_n)} f(s) = (1 \pm 0.1) \cdot \mathbf{E}[f(U_m)]] > 1 - 0.1 \cdot 2^{-(n-k)}$  for every  $f : \{0, 1\}^m \rightarrow \{0, 1\}$ , yields a  $(k, 0.2)$ -extractor  $E : \{0, 1\}^n \times \{0, 1\}^r \rightarrow \{0, 1\}^m$ . For simplicity, let us ignore the small slackness (i.e., 0.1 vs 0.2, and 2 vs 0.1) in the following discussion.

In light of the tight relationship between the extractor and the sampler in the foregoing paragraph it holds that, for  $r = O(\log n)$  and any length parameter  $m$ , having  $(k, 0.1)$ -extractors in  $\mathcal{AC}^0$  and having samplers with error  $2^{-(n-k)}$  computable in  $\mathcal{AC}^0$  is equivalent. Note that efficient error-reduction requires  $m(n) = n^{\Omega(1)}$ , since  $n$  represents the length of the input to the sampler and  $m(n)$  the length of the sampled strings, which means that  $n = \text{poly}(m(n))$  must hold.

Recall that Theorem 1.1 refers to *relatively weak extractors*; that is, ones that extract a constant factor more bits than the length of the seed (i.e.,  $m(n) \geq (1 + \Omega(1)) \cdot r(n)$ ). It asserts that  $\mathcal{AC}^0$  circuits cannot compute such extractors for min-entropy rate that is smaller than  $1/\text{poly}(\log n)$ . In contrast, recall that  $\mathcal{AC}^0[2]$  (i.e.,  $\mathcal{AC}^0$  with parity gates) circuits can compute very good extractors (e.g., Trevisan’s [54]); for example, such circuits can compute  $(k, \epsilon)$ -extractors with a logarithmically long seed for  $k(n) = m(n)^2 = \sqrt{n}$  (and  $\epsilon(n) = 1/\text{poly}(n)$ ).

Nevertheless, Theorem 1.1 says nothing about  $\mathcal{AC}^0$ -extraction from sources of higher min-entropy rate (i.e., rate at least  $1/\text{poly}(\log n)$ ). Actually, Theorem 1.2 says that  $\mathcal{AC}^0$ -extraction is possible in this case, but it only provides for extracting logarithmically many bits (i.e.,  $m(n) = O(\log n)$ ), whereas efficient error-reduction requires  $m(n) = n^{\Omega(1)}$ . Furthermore, Vadhan’s approach [55], which underlies the proof of Theorem 1.2, seems to yield  $\mathcal{AC}^0$ -extractors of logarithmic seed length only when the output length is polylogarithmic, even when the min-entropy rate is a constant. The question (see Problem 1.6) is whether one can extract more randomness under these conditions (i.e., using a source of constant min-entropy and a logarithmically long seed).

While a positive resolution regarding  $m(n) = n^{\Omega(1)}$  would imply efficient error-reduction for  $\mathcal{AC}^0$ , a bypass was found recently. Specifically, a recent revision of [30] (posted in June 2014) establishes error-reduction for  $\mathcal{AC}^0$  at the same level that would have been implied by the best  $\mathcal{AC}^0$ -extractors that are not ruled out by Theorem 1.1. That is, it offers error-reduction at a level that corresponds to min-entropy  $k(n) = n/\text{poly}(\log n)$ , output length  $m(n) = n^{\Omega(1)}$ , and logarithmic seed length (i.e.,  $r(n) = O(\log n)$ ).

This was obtained by observing that we do not really need information-theoretic extractors (computable in  $\mathcal{AC}^0$ ), but rather extractors (computable in  $\mathcal{AC}^0$ ) that output distributions

that fool all  $\mathcal{AC}^0$  circuits. Alternatively, it suffices to extract randomness for auxiliary circuits obtained by shrinking the input of the original  $\mathcal{AC}^0$  circuits by using the pseudorandom generator of Nisan [44, 45]. Hence, randomness-efficient error-reduction for  $\mathcal{AC}^0$  was obtained without presenting an  $\mathcal{AC}^0$ -extractor with the corresponding parameters. In other words, there is a gap (at least in our knowledge) between *general error-reduction implementable in  $\mathcal{AC}^0$*  (via samplers as reviewed above) and *error-reduction for  $\mathcal{AC}^0$* , which may be defined as obtaining *samplers that satisfy the sampling requirement only with respect to functions that  $f$  that are computable in  $\mathcal{AC}^0$* .

This gap in our knowledge provides a new motivation for resolving Problem 1.6, but now a resolution in the negative direction would be more interesting. Such a result would mean that, for a natural choice of parameters, randomness-efficient error-reduction for  $\mathcal{AC}^0$  exists while a corresponding  $\mathcal{AC}^0$ -extractor does not exist (where the correspondence between parameters is as in the standard relation articulated by Zuckerman [63]).

## 1.5 A roadmap and additional comments on the technical contents

Following the preliminaries, this write-up proceeds as follows. In Section 3, we prove Theorem 3.1, which is based on a non-explicit construction of Vadhan [55, Thm. 7.4]. Our improvement boils down to presenting an explicit “averaging sampler” with parameters that are comparable to the non-explicit construction. One key observation underlying the new construction is that *the relaxed notion of averaging sampler as defined by Vadhan in [55, Def. 6.1] can be composed and manipulated in ways that are not possible with the standard definition of averaging samplers*. The reason is that Vadhan’s notion is actually a hybrid of the notions of hitters and (standard) averaging samplers. Using the new sampler one can also improve the parameters in some of Vadhan’s explicit constructions of local extractors [55].

In general, Section 3 demonstrates the relevance of the study of local extractor to extraction in  $\mathcal{AC}^0$ . In particular, local extractors yield constant-depth circuits of size that is exponential in the seed length and in the locality. In some cases, the size can be made even smaller (e.g., sub-exponential in the locality).

In Section 4 we consider block-sources. In Section 4.1 we show that applying an extractor to individual blocks of a block-source, while using the same seed in all applications, yields an extractor. This result seems to be folklore, but we believe that it is a very useful one, since we think that this extraction strategy is a natural thing to do when actually having a block-source. This is relevant to the context of  $\mathcal{AC}^0$ , because the resulting extractor preserve the computational complexity of the original extractor. In Section 4.2 we consider *the difficulty of converting an arbitrary high min-entropy source into a block-source: Loosely speaking, we show that two natural approaches to this task fail*.

In Section 5 we consider  $\mathcal{AC}^0$ -extractors for bit-fixing sources. In Section 5.1, we present our first construction, which uses a randomized procedure that outputs a partition of  $[n]$  into small subsets that intersect any set of sufficient density (with high probability). The procedure uses a logarithmic amount of randomness and is explicit (and hence is implementable by uniform  $\mathcal{AC}^0$  circuits). It yields an  $\mathcal{AC}^0$ -extractor that uses a logarithmically long seed and extracts  $n/\text{poly}(\log n)$  bits from bit-fixing sources of min-entropy  $n/\text{poly}(\log n)$ . In Section 5.3 we combine the latter extractor with a new deterministic extractor (which extracts poly-logarithmically many bits), and obtain a deterministic extractor that essentially matches the performance of the seeded extractor. The former deterministic extractor is based on *a new  $\mathcal{AC}^0$ -reduction of the task of extraction from (oblivious) bit-fixing sources of entropy rate that tends to 0* (i.e.,  $1/\text{poly}(\log n)$ ) *to the task of extraction from non-oblivious bit-fixing sources of entropy rate that tends to 1* (i.e.,  $1 - 1/\text{poly}(\log n)$ ), whereas the latter



task was treated by Ajtai and Linial [4].<sup>3</sup> Indeed, the reduction is from a more restricted type of sources to a broader type of sources, but the sources of the more restricted type have significantly lower entropy.

The parameters obtained by these extractors are essentially optimal, with respect to  $\mathcal{AC}^0$ -extraction from bit-fixing sources. Indeed, in Section 5.2, we present impossibility results that extend those that are stated in Theorem 1.1:

One result asserts that that  $\mathcal{AC}^0$  circuits cannot compute a strong extractor for bit-fixing sources when the number of “unfixed” (i.e., random) bits is  $n/(\log n)^{\omega(1)}$  (regardless of the seed length). In general, Section 5.2 provides proofs of all our impossibility results, *showing that the relevant bounds hold even for extractors that should only work for bit-fixing sources.*

In Section 5.4 we show that the foregoing impossibility results regarding extraction from bit-fixing sources do not hold for a restricted class of such sources, called zero-fixing sources [19], consisting of bit-fixing sources in which all fixed bits are set to zero. We show that  $\mathcal{AC}^0$  circuits, which use a seed of logarithmic length, can extract from zero-fixing sources that contain only a logarithmic number of random bits.

Turning back to general sources of min-entropy  $k$ , in Section 6 we consider extraction with a seed of linear length. In Section 6.1 we generalize Viola’s construction [58, Lem. 4.3] of a non-trivial extractor in  $\mathcal{AC}^0$  to one that outputs  $\text{poly}(\log n)$  additional bits (rather than one). In Section 6.2 we show that independent applications of an extractor (i.e., with independently distributed seeds) yield an extractor, provided that the total number of extracted bits does not exceed the min-entropy bound. This is a naive result, which relies on well-known ideas, but it is advantageous in the context of  $\mathcal{AC}^0$  because the resulting extractor preserve the computational complexity of the original extractor. Indeed, this simple observation is pivotal in establishing the general upper bound of Theorem 1.7, which presents a trade-off between the seed length and the number of (additional) bits extracted.

In Section 7, we consider deterministic  $\mathcal{AC}^0$ -extractor for several independent sources. Leaving open the question of extraction from *pairs* of sources of some constant min-entropy rate, we prove the existence of  $\mathcal{AC}^0$ -extractors for pairs of sources of min-entropy rate  $1 - \log^{-4} n$ . This is obtained by presenting a *uniform  $\mathcal{AC}^0$ -reduction of the task at hand to the task of extraction from non-oblivious bit-fixing sources of entropy rate  $1 - O(\log^{-3} n)$* . (Unlike in Section 5.3 here the reduction is between incomparable types of sources and the target sources have lower entropy.) We also present, for any constant  $\delta > 0$ , *explicit  $\mathcal{AC}^0$ -extractors for  $\text{poly}(1/\delta)$  independent sources that are each of min-entropy rate  $\delta$ , whereas for  $\delta = 0.99$  four sources suffice.*

Finally, in Section 8, we list and restate open problems that are mentioned in various parts of the paper.

## 2 Preliminaries

By “circuits” we mean Boolean circuits with negations and unbounded fan-in **and** and **or** gates. Since we deal with a very low complexity class (i.e.,  $\mathcal{AC}^0$ ), it is important to be careful about the representation of objects. In particular, elements of  $[n] = \{1, 2, \dots, n\}$  are represented as  $(\lceil \log_2 n \rceil + 1)$ -bit long strings, and subsets of  $[n]$  are represented as (unsorted)

<sup>3</sup> In non-oblivious bit-fixing sources the fixed bits may be determined as a function of the values of the random bits, whereas in (oblivious) bit-fixing sources the fixed bits are set independently of the values of the random bits. The connection between influence of sets as studies in [4] and deterministic extraction from non-oblivious bit-fixing sources was made in [41].



sequences over  $[n]$ . This convention is important for supporting an efficient implementation of the projection operation; that is, for a sequence  $x = (x_1, \dots, x_n) \in \{0, 1\}^n$  and  $I \subseteq [n]$ , we let  $X_I$  denote the projection of  $x$  on  $I$  (i.e., if  $I = (i_1, \dots, i_t)$ , then  $X_I = (x_{i_1}, \dots, x_{i_t})$ ). Indeed, the mapping  $(x, (i_1, \dots, i_t)) \mapsto x_{(i_1, \dots, i_t)}$  is computable in uniform  $\mathcal{AC}^0$  (e.g., by having the  $j^{\text{th}}$  output bit equal  $\bigvee_{i \in [n]} (x_i \wedge (i = i_j))$ ).<sup>4</sup>

By  $\log n$  we mean  $\log_2 n$ , whereas by  $\text{poly}(n)$  we mean any (unspecified) positive polynomial. In several statements (e.g., Theorem 1.1), we preferred to use 0.999 (resp., 0.499) rather than “for every constant smaller than one” (resp., “for every constant smaller than half”).

Another general convention is that multiple occurrences of the same random variable mean that the same random value is assigned in all occurrences; that is, if  $X$  is a random variable, then  $(X, X, X)$  represents the random variable obtained by selecting  $x$  according to  $X$  and outputting  $(x, x, x)$ . By  $U_\ell$  we denote a random variable that is uniformly distributed over  $\{0, 1\}^\ell$ .

Below we review the basic definitions regarding extractors as well as some results regarding locally computable extractors.

## 2.1 General extractors

We recommend Shaltiel’s survey [53] for a general introduction to randomness extractors.

► **Definition 2.1** (min-entropy and  $(n, k)$ -sources). The min-entropy of a random variable  $X$ , denoted  $\mathbf{H}_\infty(X)$ , is  $\min_x \{\log_2(1/\mathbf{Pr}[X = x])\}$ . An  $(n, k)$ -source is a random variable  $X$  assuming values in  $\{0, 1\}^n$  such that  $\mathbf{H}_\infty(X) \geq k$ .

(Indeed,  $\max_x \{\mathbf{Pr}[X = x]\} \leq 2^{-\mathbf{H}_\infty(X)}$ .) In the following definition,  $\Delta[X; Y]$  denotes the statistical difference (a.k.a variation distance) between  $X$  and  $Y$ ; that is,

$$\Delta[X; Y] \stackrel{\text{def}}{=} \frac{1}{2} \cdot \sum_v |\mathbf{Pr}[X = v] - \mathbf{Pr}[Y = v]| = \max_S \{\mathbf{Pr}[X \in S] - \mathbf{Pr}[Y \in S]\}$$

► **Definition 2.2** ((seeded) randomness extractors). The function  $E : \{0, 1\}^n \times \{0, 1\}^r \rightarrow \{0, 1\}^m$  is called an  $\epsilon$ -error extractor for a class of sources  $\mathcal{C}$  if for every  $X$  in  $\mathcal{C}$  it holds that  $\Delta[E(X, U_r); U_m] \leq \epsilon$ . It is called a strong  $\epsilon$ -error extractor for  $\mathcal{C}$  if for every  $X$  in  $\mathcal{C}$  it holds that  $\Delta[E(X, U_r) \circ U_r; U_m \circ U_r] \leq \epsilon$ . When  $\mathcal{C}$  is the class of  $(n, k)$ -sources, we call  $E$  a  $(k, \epsilon)$ -extractor.

Note that

$$\Delta[E(X, U_r) \circ U_r; U_m \circ U_r] = \mathbf{E}_{s \leftarrow U_r} [\Delta[E(X, s); U_m]].$$

We say that an extractor is explicit if a circuit computing it can be constructed in  $\text{poly}(n)$ -time. Indeed, to make sense of this definition, one should consider a family of extractors parameterized by  $n$ ; that is, we actually consider the family  $\{E_n : \{0, 1\}^n \times \{0, 1\}^{r(n)} \rightarrow \{0, 1\}^{m(n)}\}_{n \in \mathbb{N}}$  such that  $E_n$  is an  $(k(n), \epsilon(n))$ -extractor. Likewise, when using asymptotic notation in reference to an extractor  $E : \{0, 1\}^n \times \{0, 1\}^r \rightarrow \{0, 1\}^m$ , we actually refer to a

<sup>4</sup> In contrast, if  $I \subseteq [n]$  is represented by the  $n$ -bit long indicator vector  $\chi = (\chi_1, \dots, \chi_n)$  such that  $\chi_i = 1$  if  $i \in I$  and  $\chi_i = 0$  otherwise, then the mapping  $(x, I) \mapsto x_I$  is not computable in  $\mathcal{AC}^0$ . The reason is that counting (i.e., computing  $\sum_{i \in [n]} b_i$ ) is  $\mathcal{AC}^0$ -reducible to the foregoing operation by setting  $x = 1^n$  and measuring the length of  $x_I$ , where  $I = \{i \in [n] : b_i = 1\}$ , while relying on  $(b_1, \dots, b_n)$  being an admissible representation of  $I$ .

family as above; that is, we always view the source length (i.e.,  $n$ ) as a varying parameter (which determines all other parameters; e.g.,  $r = r(n)$ ,  $m = m(n)$ , etc). Furthermore, we assume that  $k, r, m : \mathbb{N} \rightarrow \mathbb{N}$  are monotonically non-decreasing and that  $\epsilon : \mathbb{N} \rightarrow [0, 1]$  is monotonically non-increasing. These conventions are used extensively throughout this write-up.

## 2.2 Local extractors

Locally computable extractors were systematically studied by Vadhan [55]. These constructs as well as some of Vadhan's techniques (see especially [55, Sec. 6]) are used by us towards constructing extractors that can be computed in  $\mathcal{AC}^0$ .

► **Definition 2.3** (*t*-local extractors). The extractor  $E : \{0, 1\}^n \times \{0, 1\}^r \rightarrow \{0, 1\}^m$  is called *t*-local if of every  $s \in \{0, 1\}^r$  the residual function  $f_s(x) \stackrel{\text{def}}{=} E(x, s)$  depends on at most  $t$  bits of  $x$ .

For the case of constant min-entropy rate, we have the following result.

► **Theorem 2.4** (special case of [55, Thm. 8.5]). *For every constants  $\delta, \beta > 0$  and  $\epsilon(n) = 1/\text{poly}(n)$ , there exists an explicit  $O(\log n)$ -local strong  $(\delta n, \epsilon(n))$ -extractor  $E : \{0, 1\}^n \times \{0, 1\}^{\beta \cdot m(n)} \rightarrow \{0, 1\}^{m(n)}$  for  $m(n) = \Theta(\log n)$ . Furthermore, the extractor is computable in uniform  $\mathcal{AC}^0$ .*

The furthermore claim is not stated in [55], but it follows from the main claim by combining the circuits that compute the residual functions (obtained by fixing the seed). Specifically, we combine depth-two circuits that compute the residual functions (where each function depends on  $O(\log n)$  bits), with depth-two circuits that select the adequate function, obtaining depth-three circuits. For the case of min-entropy rate  $1/\text{poly}(\log n)$ , we mention the following non-explicit construction.

► **Theorem 2.5** (special case of [55, Thm. 7.4]). *For every  $k(n) = n/\text{poly}(\log n)$  and  $\epsilon(n) = 1/\text{poly}(n)$ , there exists an (non-explicit)  $(2n/k(n))$ -local strong  $(k(n), \epsilon(n))$ -extractor  $E : \{0, 1\}^n \times \{0, 1\}^{O(\log n)} \rightarrow \{0, 1\}^{\Omega(\log n)}$ .*

(Note that here the seed is longer than the output, but the result is non-trivial since the extractor is strong.) While the relevance of Theorem 2.4 to our study was demonstrated in its furthermore-part, the relevance of Theorem 2.5 is less clear and arises from the technique used in its proof. The point is that the proof of Theorem 2.5 is based on the composition theorem of [55, Thm. 6.3], which implies that combining an adequate sampler with an adequate extractor yields an extractor that can be computed *more efficiently than standard extractors*. Loosely speaking, the sampler is used to sample relatively few bits in the source, and the extractor is applied to the resulting sequence of bits, which approximately maintains the entropy rate of the source. Thus, the complexity of the resulting extractor is related to the complexity of sampling and to the complexity of extraction from a much shorter source. (This fact will be extensively used in Section 3.)

The sample-and-extract paradigm goes back to the work Nisan and Zuckerman [46], but the point here is using it in order to reduce the computational complexity of extraction.

Furthermore, better parameters are obtained by using the following relaxed notion of an averaging sampler, introduced by Vadhan [55], which is a hybrid of a sampler and a hitter (see discussion following the definition).

► **Definition 2.6** (averaging samplers, relaxed [55, Def. 6.1]<sup>5</sup>). A function  $S : \{0, 1\}^r \rightarrow [n]^t$  is called a  $(\mu, \mu', \gamma)$ -averaging sampler if for every  $f : [n] \rightarrow [0, 1]$  such that  $\rho(f) \stackrel{\text{def}}{=} \mathbf{E}_{i \in [n]}[f(i)] \geq \mu$  it holds that

$$\Pr_{I \leftarrow S(U_r)} \left[ \frac{1}{t} \sum_{i \in I} f(i) < \mu' \right] \leq \gamma. \quad (1)$$

Furthermore, it is required that  $|S(u)| = t$  for every  $u \in \{0, 1\}^r$ ; that is, the sampler must always generate  $t$  *distinct* elements.

The standard notion of a  $(\delta, \gamma)$ -averaging sampler is obtained from Definition 2.6 by requiring that  $S$  is an  $(\mu, \mu - \delta, \gamma)$ -averaging sampler for every  $\mu \in [0, 1]$ . (Note that in such a case an upper bound on the probability that  $\frac{1}{t} \sum_{i \in I} f(i) > \rho(f) + \delta$  follows by considering the function  $1 - f$ .) The definition of a  $(\mu, \gamma)$ -hitter is obtained from Definition 2.6 by replacing Eq. (1) with  $\Pr_{I \leftarrow S(U_r)}[\sum_{i \in I} f(i) = 0] \leq \gamma$ . (Indeed, the latter definition remains intact if one only considers Boolean functions  $f : [n] \rightarrow \{0, 1\}$  (such that  $\rho(f) \geq \mu$ ).)<sup>6</sup> (In Appendix A.1 we prove two useful features of such averaging samplers, although these features are not essential to this write-up.)

The relevance of averaging samplers to our project is captured by the following composition theorem of Vadhan [55].

► **Theorem 2.7** (sample-then-extract [55, Thm. 6.3]<sup>7</sup>). *For any  $0 < 3\tau < \delta \leq 1$ , let  $\mu = (\delta - 2\tau)/\log(1/\tau)$  and  $\mu' = (\delta - 3\tau)/\log(1/\tau)$ . Suppose that the following two conditions hold.*

1.  $S : \{0, 1\}^r \rightarrow [n]^t$  is a  $(\mu, \mu', \gamma)$ -averaging sampler;
  2.  $E_0 : \{0, 1\}^t \times \{0, 1\}^{r_0} \rightarrow \{0, 1\}^m$  is a  $((\delta - 3\tau) \cdot t, \epsilon_0)$ -extractor.
- Then,  $E : \{0, 1\}^n \times \{0, 1\}^{r_0+r} \rightarrow \{0, 1\}^m$  defined by  $E(x, (s_0, s)) = E_0(x_{S(s)}, s_0)$  is a  $(\delta \cdot n, \epsilon_0 + \gamma + \exp(-\Omega(\tau n)))$ -extractor. Furthermore, if  $E_0$  is strong then so is  $E$ .*

For any  $\beta \in (0, 1)$ , setting  $\tau = (1 - \beta)\delta/3$  and assuming that  $\delta = \omega(\log(1/\epsilon)/n)$ , we obtain the following simplified form.

► **Corollary 2.8** (Theorem 2.7, specialized). *For any  $\beta \in (0, 1)$  and  $\delta = \omega(\log(1/\epsilon)/n)$ , let  $\mu' = \Theta(\delta/\log(1/\delta))$  and  $\mu = \frac{1+2\beta}{3\beta} \cdot \mu' = (1 + \Omega(1)) \cdot \mu'$ . Suppose that the following two conditions hold.*

1.  $S : \{0, 1\}^r \rightarrow [n]^t$  is a  $(\mu, \mu', \epsilon)$ -averaging sampler;
  2.  $E_0 : \{0, 1\}^t \times \{0, 1\}^{r_0} \rightarrow \{0, 1\}^m$  is a  $(\beta\delta \cdot t, \epsilon)$ -extractor.
- Then,  $E : \{0, 1\}^n \times \{0, 1\}^{r_0+r} \rightarrow \{0, 1\}^m$  defined by  $E(x, (s_0, s)) = E_0(x_{S(s)}, s_0)$  is a  $(\delta \cdot n, 3\epsilon)$ -extractor. Alternatively, if  $\delta = \Omega(\log(1/\gamma)/n)$  and  $S$  is a  $(\mu, \mu', \gamma)$ -averaging sampler, then  $E$  is a  $(\delta \cdot n, \epsilon + \gamma^{\Omega(1)})$ -extractor. Furthermore, if  $E_0$  is strong then so is  $E$ .*

Indeed, when using Corollary 2.8 one should artificially set the error parameter of both constructs in the hypothesis to be the maximum of their actual values. Note that *if both  $S$  and  $E_0$  are computable in (uniform)  $\mathcal{AC}^0$ , then so is  $E$ .*

<sup>5</sup> The formulation in [55, Def. 6.1] is slightly different: Firstly, the current  $(\mu, \mu', \gamma)$ -averaging sampler corresponds to a  $(\mu, \mu - \mu', \gamma)$ -averaging sampler in [55, Def. 6.1]. Secondly, the “distinct element condition” is an integral part of Definition 2.6, whereas it is an additional feature in [55, Def. 6.1].

<sup>6</sup> See the proof of [28, Thm. 5.10], which is adapted in the proof of Claim A.2.

<sup>7</sup> The formulation in [55, Thm. 6.3] is slightly different: See Footnote 5.

### 3 Local extractors and extraction in $\mathcal{AC}^0$

In this section, we use local extractors and the ideas underlying their construction to obtain extractors computable in  $\mathcal{AC}^0$ . This approach was already used in proving Theorem 2.4, and we stated our intention to use it towards proving Theorems 1.2 and 3.1. Let us spell out how Theorem 2.4 was proved, in order to clarify the connection between local extraction and extraction in  $\mathcal{AC}^0$ .

Theorem 2.4 asserts an (explicit) extractor of logarithmic locality and logarithmic seed length. This means that for any possible seed, the residual extraction function depends only on logarithmically many bits in the source, which implies that these residual functions can be computed by depth-two circuits of polynomial size. Combining the polynomially many circuits that correspond to all possible fixing of the seed, we obtain the desired  $\mathcal{AC}^0$ -extractor.

Note that the foregoing argument can be turned around: It implies that if  $\mathcal{AC}^0$  cannot compute extractors with logarithmic seed length for certain parameters, then such extractors cannot have logarithmic locality. We mention that Bogdanov and Guo [11] proved a logarithmic lower bound on the locality of extractors, which (unlike our lower bounds) applies also to the case of  $k(n) = \Omega(n)$ , but this does not rule out  $\mathcal{AC}^0$ -extractors (see Theorem 2.4).

Turning to Theorem 1.2 and its explicit version (captured by Theorem 3.1), recall that its proof is based on Vadhan's sample-then-extract technique (as stated in [55, Thm. 6.3] and restated in Corollary 2.8). The starting point is the proof of Theorem 2.5, which uses a sampler (of a logarithmically long seed) that samples poly-logarithmically many bits of the source. Unlike Vadhan, we cannot afford an arbitrary extractor here, so instead we use an extractor that can be computed by (explicit) constant-depth circuits of poly( $n$ )-size (see Section 3.1 for details). In addition, we make the entire construction explicit by using a new explicit sampler (instead of the non-explicit sampler used originally in [55]). Thus, we obtain:

► **Theorem 3.1** (an explicit version of Theorem 1.2). *For every  $k(n) = n/\text{poly}(\log n)$  and  $\epsilon(n) = 1/\text{poly}(n)$ , there exist explicit  $\mathcal{AC}^0$  circuits that compute a strong  $(k(n), \epsilon(n))$ -extractor  $E : \{0, 1\}^n \times \{0, 1\}^{O(\log n)} \rightarrow \{0, 1\}^{\Theta(\log n)}$ . Furthermore, the circuits have depth  $4 + \left\lceil \frac{\log(n/k(n))}{\log \log n} \right\rceil$ .*

The new explicit sampler, presented in Section 3.2, uses a seed of logarithmic length, and so it is trivially implemented by uniform  $\mathcal{AC}^0$  circuits. As stated in Corollary 3.5, combining this sampler with Corollary 2.8, the construction of  $\mathcal{AC}^0$ -extractors for  $(n, k)$ -sources reduces to the construction of poly( $n$ )-circuits of constant depth for extraction from  $(t, \Omega(k/n) \cdot t)$ -sources, where  $t = \text{poly}(n/k)$ . Since  $k = n/\text{poly}(\log n)$ , we can afford constant-depth circuits of size  $\exp(t^c)$  for a sufficiently small constant  $c > 0$ .

The applications of the new sampler are spelled out in Section 3.3. Since the new extractor works only for  $t = \tilde{O}(n^{1/3})$  (or for constant error  $\gamma > 0$ ), it does not suffice for the application in Section 6, and so we present (in Section 3.4) an alternative sampler that works essentially for any  $t$  and any error  $\gamma = 1/\text{poly}(n)$ , which suffices for the latter application. The alternative version uses a seed of polylogarithmic length, which forces us to detail its implementation in  $\mathcal{AC}^0$ .

#### 3.1 Proving Theorem 1.2

As a warm-up, we prove Theorem 1.2, which asserts that *for every  $k(n) = n/\text{poly}(\log n)$  and  $\epsilon(n) = 1/\text{poly}(n)$ , there exist  $\mathcal{AC}^0$  circuits that compute a strong  $(k(n), \epsilon(n))$ -extractor  $E : \{0, 1\}^n \times \{0, 1\}^{O(\log n)} \rightarrow \{0, 1\}^{O(\log n)}$ . (Furthermore, the circuits have depth  $4 + \frac{\log(n/k(n))}{\log \log n}$ .)*

**Proof Sketch.** We slightly modify the proof of [55, Thm. 7.4], which combines an averaging sampler with an extractor (using [55, Thm. 6.3] (see Corollary 2.8)), while setting  $\delta = k(n)/n = 1/\text{poly}(\log n)$ ,  $m = \Theta(\log n)$  and  $t = O(m/\delta)$ . Specifically, instead of using the (optimal) non-explicit extractor asserted in [55, Lem. 7.1], which may not be computable in  $\mathcal{AC}^0$ , we shall use the strong  $(k_0, \epsilon)$ -extractor  $E_0 : \{0, 1\}^t \times \{0, 1\}^{O(\log n)} \rightarrow \{0, 1\}^{m(n)}$  asserted in [29, Sec. 5], where  $k_0 = O(m + \log(1/\epsilon))$ . Recall that this extractor uses a seed of length  $O(m + \log(t/\epsilon)) = O(\log n)$ . (At this point we use the same non-explicit averaging sampler as in the original proof; that is, the sampler of [55, Lem. 7.2].)

The key observation is that  $E_0$  can be computed by (uniform)  $\text{poly}(n)$ -size circuits of depth  $2 + \log_m t = 3 + \frac{\log(n/k(n))}{\log m}$ , since the residual computation (resulting when fixing the seed) amounts to a linear combination (over  $\text{GF}(2^{O(\log t)})$ ) of  $O(\log t)$ -bit long blocks of the  $t$ -bit source viewed as a sequence of  $t/O(\log t)$  elements over a field of size  $\text{poly}(t)$ . The sampler itself uses a logarithmically long seed, and thus can be computed by depth two circuits of polynomial size. ◀

### Towards the explicit version

The foregoing proof, which follows the proof of [55, Lem. 7.1], combines an adequate sampler with an adequate extractor using the composition theorem of [55, Thm. 6.3]. Recall that we replaced the extractor used in the original proof by an alternative extractor, which happens to be explicit. In order to obtain an explicit version of the foregoing construction (i.e., prove an explicit version of Theorem 1.2), we also need to replace the non-explicit sampler used in the original proof. This will be done in Section 3.3, after designing an explicit sampler (in Section 3.2).

## 3.2 A new averaging sampler

As stated in Section 3.1, a central ingredient in our constructions is a new sampler that uses a seed of logarithmic length regardless of the density (i.e.,  $\mu$ ) of the functions that it semi-approximates, where by “semi-approximates” we refer to the relaxed notion of an averaging sampler (as reviewed in Definition 2.6).

► **Theorem 3.2** (an averaging sampler with logarithmic seed length). *Let  $\alpha \in (0, 1)$  be an arbitrary constant. Then, for every  $\mu > n^{-1/3}/\text{poly}(\log n)$  and  $\gamma \geq 1/\text{poly}(n)$ , there exists an explicit  $(\mu, \alpha\mu, \gamma)$ -averaging sampler  $S : \{0, 1\}^{O(\log n)} \rightarrow [n]^t$  for any  $t \in [\Theta(\mu^{-1} \log(n/\gamma)), \tilde{O}(n^{1/3})]$ . An analogous result holds for any  $\mu > 1/n$ ,  $t = \Omega(1/\mu)$ , and any constant  $\gamma > 0$ .*

That is, the main claim requires  $\mu > n^{-1/3}$  and allows  $\gamma \geq 1/\text{poly}(n)$ , whereas the alternative allows any  $\mu > 0$  (equiv.,  $\mu > 1/n$ ) but provide only for constant  $\gamma > 0$ . (The constant  $1/3$ , in the exponent, can be replaced by any constant smaller than  $1/2$ .)

**Proof.** We first establish the result for a constant  $\gamma$ , and then extend it to arbitrary  $\gamma \geq 1/\text{poly}(n)$ . Our starting point is the observation that the pairwise-independence generator yields a  $(\mu, \alpha\mu, \gamma)$ -averaging sampler with  $t = \Theta(1/\gamma\mu)$  samples. Basically, the behavior of this generator with respect to the relaxed notion of averaging samplers (as in Definition 2.6) is similar to its behavior with respect to the notion of hitters (i.e., the sample size is linearly related to  $O(1/\mu)$ ).

► **Claim 3.3** (the pairwise independence generator as an averaging sampler). *Let  $F$  be a finite field,  $t < |F|$ , and  $\phi_1, \dots, \phi_t$  be  $t$  distinct non-zero elements of  $F$ . Consider  $G : F^2 \rightarrow F^t$*

such that, for every  $r, s \in F$ ,

$$G(r, s) = (r + \phi_1 s, r + \phi_2 s, \dots, r + \phi_t s).$$

Then, for any constants  $\alpha, \beta \in (0, 1)$ , selecting uniformly and independently  $r \in F$  and  $s \in F \setminus \{0\}$ , and outputting  $G(r, s)$  yields a  $(\mu, \alpha\mu, \beta)$ -averaging sampler for  $\mu = \Omega(1/t)$ .

We shall also use the fact that each of the elements in the sample is uniformly distributed in  $F$ . An alternative construction that enjoys all the above features is the expander neighborhood generator (see, e.g., [28, Apdx. C.2]), when instantiated with Ramanujan graphs (see [42]).

**Proof.** The proof is by a straightforward adaptation of the proof of the hitting property (see, e.g., [28, Apdx. C.2]). Specifically, for  $r$  and  $s$  selected uniformly in  $F$ , and any function  $f : F \rightarrow [0, 1]$  such that  $\rho(f) \geq \mu$ , denoting by  $\zeta_j$  the value of  $f(r + \phi_j s)$ , we have

$$\begin{aligned} \Pr \left[ \sum_{j=1}^t \zeta_j < \alpha \cdot \rho(f) \cdot t \right] &\leq \Pr \left[ \left| t \cdot \rho(f) - \sum_{j=1}^t \zeta_j \right| > (1 - \alpha) \cdot t \cdot \rho(f) \right] \\ &\leq \frac{t \cdot (1 - \rho(f)) \cdot \rho(f)}{((1 - \alpha) \cdot t \cdot \rho(f))^2} \\ &< \frac{1}{(1 - \alpha)^2 \cdot \mu \cdot t} \end{aligned}$$

where the second inequality uses Chebyshev's Inequality. Avoiding the choice of  $s = 0$  guarantees that the  $t$  elements in  $G(r, s)$  are indeed disjoint, while skewing the probability by at most  $1/|F| = o(1)$ . ◀

**Establishing the alternative claim (of the theorem) and moving on.** Associating  $[n]$  with the non-zero elements of a finite field (of size approximately  $n$ )<sup>8</sup> and using any  $t \geq O(1/\mu)$ , we obtain an  $(\mu, \alpha\mu, \beta)$ -averaging sampler that uses a seed of length  $2 \log n$  (as in the alternative claim).

In order to reduce the error probability to  $\gamma = 1/\text{poly}(n)$ , we shall use a random walk of length  $\ell = O(\log(n/\gamma))$  on a constant-degree expander over the vertex set  $[n^2]$  to generate seeds for the generator  $G$  of Claim 3.3. Actually, in order to guarantee that these seeds do not generate samples that intersect, we will shift these seeds using a  $O(1)$ -wise independent sequence over  $[n]^2$ , and discard a small part of the resulting sample. The point is that this combination yields a sequence that maintains the sampling features of the expander random walk and the property that each  $O(1)$  elements of the sequence are independently and uniformly distributed in the set. (At this point, we shall discard the few repetitions.)

► **Claim 3.4** (a randomized version of the “random walk on an expander” hitter). *For every constants  $\epsilon > 0$  and  $c \in \mathbb{N}$  and a varying  $m$ , let  $\ell = O(\log m)$  be sufficiently large, and consider the following generator, denoted  $G'$ , that uses a seed of length  $O(\log m)$ .*

1. *The generator uses the first part of the seed to generate a random walk of length  $\ell$  on a  $\text{poly}(1/\epsilon)$ -regular expander with vertex set  $\mathbb{Z}_m$ . Denote the sequence of visited vertices by  $(v_1, \dots, v_\ell)$ .*
2. *The generator uses the second part of the seed to generate a  $2c$ -wise independent sequence of  $\ell$  elements in  $\mathbb{Z}_m$ , denoted  $(s_1, \dots, s_\ell)$ .*

---

<sup>8</sup> The discrepancy between the field size and  $n$  can be ignored (e.g., by using a slightly bigger field and mapping elements out of  $[n]$  in some standard manner).



3. For every  $i \in [\ell]$ , the generator computes  $v'_i \leftarrow v_i + s_i \pmod m$ .
  4. The generator outputs the sequence of the first  $\ell' = \ell - c$  distinct elements in the sequence  $(v'_1, \dots, v'_\ell)$ , and outputs  $(1, \dots, \ell')$  if  $|\{v'_i : i \in [\ell]\}| < \ell - c$ .
- Then, the foregoing generator is a  $(1 - \epsilon, 1 - O(\sqrt{\epsilon}), \exp(-\Omega(\ell))$ -averaging sampler.

In other words, for every constants  $\epsilon > 0$  and  $c \in \mathbb{N}$ , the generator  $G' : \{0, 1\}^{O(\log m)} \rightarrow (\mathbb{Z}_m)^{O(\log m)}$  is a  $(1 - \epsilon, 1 - O(\sqrt{\epsilon}), m^{-c})$ -averaging sampler.

**Proof.** We first upper bound the probability that  $|\{v'_i : i \in [\ell]\}| < \ell - c$ . Fixing any sequence of  $v_i$ 's (as selected in Step 1), we consider the probability space generated by Step 3. For the bad event to occur, there must be a set of  $2c$  indices, denoted  $I$ , such that the set  $S_I \stackrel{\text{def}}{=} \{v_i + s_i \pmod m : i \in I\}$  has cardinality smaller than  $c$ . Since for every  $2c$ -subset  $I$  it holds that  $\Pr[|S_I| < c] < \binom{m}{c} \cdot (c/m)^{2c} < (c^2/m)^c$ , the bad event occurs with probability at most  $\binom{\ell}{2c} \cdot (c^2/m)^c < (c^2 \ell^2/m)^c < 2^{-\Omega(\ell)}$ , provided that  $\ell/\log m$  is a sufficiently large constant.

We now analyze the sampling property of the sequence of all  $v'_i$ 's, while noting that any subsequence of length  $\ell'$  will do almost as well (since the omission causes a loss of at most  $c = o(\sqrt{\epsilon} \ell)$  units). Fixing an arbitrary function  $f : \mathbb{Z}_m \rightarrow [0, 1]$  such that  $\rho(f) \geq 1 - \epsilon$ , we now fix an arbitrary sequence  $(s_1, \dots, s_\ell)$  as selected in Step 2. This selection induces  $\ell$  auxiliary functions  $f_1, \dots, f_\ell$  such that  $f_i(x) = f(x + s_i \pmod m)$ , since under this definition  $f(v'_i) = f_i(v_i)$ . Setting  $\epsilon' = 6\sqrt{\epsilon}$ , we shall prove that

$$\Pr \left[ \sum_{i \in [\ell]} f_i(v_i) < \ell \cdot (1 - \epsilon') \right] = \exp(-\Omega(\ell)), \tag{2}$$

where  $(v_1, \dots, v_\ell)$  is generated as in Step 1. We recall the *general* Expander Random Walk Theorem (see [28, Thm. A.4]), which asserts that for Boolean functions  $b_i : [m] \rightarrow \{0, 1\}$  it holds that  $\Pr[\sum_{i \in [\ell]} b_i(v_i) = \ell] < \prod_{i \in [\ell]} \min(1, \rho(b_i) + \epsilon)^{1/2}$ , where  $\epsilon$  upper bounds the (square of) the spectral gap of the expander (i.e., the expander was chosen so that this upper bound holds). Hence, Eq. (2) follows for *Boolean*  $f_i$ 's by considering all  $\epsilon' \ell$ -subsets  $I \subset [\ell]$  and setting  $b_i = 1 - f_i$  if  $i \in I$  and  $b_i = 1$  otherwise. Specifically, the probability that  $\sum_{i \in [\ell]} f_i(v_i) < (1 - \epsilon') \cdot \ell$  is upper bounded by  $\binom{\ell}{\epsilon' \ell} \cdot (2\epsilon)^{\epsilon' \ell/2}$ , since this event can occur only if there exists a  $\epsilon' \ell$ -subset  $I$  such that for every  $i \in I$  it holds that  $f_i(v_i) = 0$  (whereas  $\rho(f_i) \geq 1 - \epsilon$ ).<sup>9</sup> Now, using

$$\begin{aligned} \binom{\ell}{\epsilon' \ell} \cdot (2\epsilon)^{\epsilon' \ell/2} &< (3\ell/\epsilon' \ell)^{\epsilon' \ell} \cdot (2\epsilon)^{\epsilon' \ell/2} \\ &= (1/2)^{\epsilon' \ell/2}, \end{aligned}$$

where the equality uses  $\epsilon' = 6\sqrt{\epsilon}$ , and the claim follows (for Boolean  $f$ ).

Observing that any averaging sampler for Boolean functions also works for general functions (see Claim A.2), the claim is established. In this case, we can establish that the generator is a  $(1 - 0.5\epsilon, 1 - O(\sqrt{\epsilon}), \exp(-\Omega(\ell))$ -averaging sampler. An alternative argument for the current setting may proceed by defining auxiliary Boolean functions  $f'_i$  such that  $f'_i(x) = 1$  if and only if  $f_i(x) \geq 1 - \epsilon^{1/3}$ , noting that  $\rho(f'_i) \geq 1 - \epsilon^{2/3}$ , and using  $\epsilon' = O(\epsilon^{1/3})$ . (This

<sup>9</sup> Note that, for this  $I$ , we have  $\rho(b_i) \leq \epsilon$  for  $i \in I$  and  $b_i \equiv 1$  otherwise. Hence,  $\sum_{i \in I} f_i(v_i) = 0$  if and only if  $\sum_{i \in I} b_i(v_i) = |I|$ , which holds if and only if  $\sum_{i \in [\ell]} b_i(v_i) = \ell$ , whereas the probability of the latter event is upper bounded by  $\prod_{i \in [\ell]} \min(1, \rho(b_i) + \epsilon)^{1/2} \leq (2\epsilon)^{|I|/2}$ .



alternative only establishes that the generator is a  $(1 - \epsilon, 1 - O(\epsilon^{1/3}), \exp(-\Omega(\ell)))$ -averaging sampler, which is good enough for our application.  $\blacktriangleleft$

**Establishing the main claim of the theorem.** The theorem follows by combining the samplers asserted in Claims 3.3 and 3.4. Specifically, we use  $G'$  of Claim 3.4 to generate a sequence of  $\ell'$  seeds for the generator  $G$  of Claim 3.3 (i.e., we set  $m = n^2$  and associate  $\mathbb{Z}_m$  with  $(F \setminus \{0\})^2 \equiv [n]^2$ ). Actually, we might as well use the sequence of all  $\ell$  seeds defined in Step 3 of  $G'$  (since we are going to make our own omissions anyhow). We set the constant parameter  $\beta$  in Claim 3.3 so to fit  $\epsilon$  in Claim 3.4 (i.e.,  $\beta = \epsilon$ ), while picking the constant parameter  $\alpha < 1$  as large as we wish (and ditto w.r.t  $t \in [\Theta(\mu^{-1} \log(n/\gamma)), \tilde{O}(n^{1/3})]$ ). That is, the combined generator maps its seed  $s \in \{0, 1\}^{O(\log n)}$  to the sequence of sets  $G(s_1), \dots, G(s_\ell)$ , where  $(s_1, \dots, s_\ell) \leftarrow G'(s)$ .

The combined generator satisfies the average sampling property of Eq. (1); that is, for every  $f : [n] \rightarrow [0, 1]$  such that  $\rho(f) \geq \mu$ , with probability at least  $1 - \gamma$ , the  $\ell \cdot t$ -sized sample, denoted  $I$ , satisfies  $(1/|I|) \cdot \sum_{i \in I} f(i) \geq (1 - O(\sqrt{\epsilon})) \cdot \alpha \mu$ . This is the case since at least  $1 - O(\sqrt{\epsilon})$  fraction of the seeds generated by  $G'$  produce samples that have an  $f$ -average of at least  $\alpha \mu$ . (Indeed, the one-sided flavor of Eq. (1) allows to discard the exceptional seeds.) Since  $\epsilon > 0$  and  $1 - \alpha > 0$  can be made arbitrarily small constants, we can have  $(1 - O(\sqrt{\epsilon})) \cdot \alpha$  be arbitrarily close to one. We note that the sampling guarantee remains valid also if we omit any  $o(\ell)$  of the  $\ell$  samples (produced by the  $\ell$  seeds generated by  $G'$ ).

The question is whether the sample  $I$  contains no repetitions. Recall that  $G$  generates samples that have no repetitions, and that the  $t$  elements in each sample are each uniformly distributed in  $[n]$ . For any desired  $c$ , the seeds generated by  $G'$  are  $2c$ -wise independent. Now, consider a random graph  $R$  with vertices corresponding to the  $\ell$  seeds and edges connecting a pair of seeds if and only if the corresponding samples intersect. We claim that, with probability at least  $1 - (\ell^2 t^2/n)^c$ , this graph has an independent set of size  $\ell - 2c$ .

To prove the last claim note that if the graph has no vertex cover of size  $2c$ , then it must have a matching of size greater than  $c$ . We shall show that even having a matching of size  $c$  is unlikely. Denoting the number of possible matching of size  $c$  in an  $\ell$ -vertex graph by  $M < \binom{\ell}{2c} \cdot (2c!) < \ell^{2c}$ , note that the probability that the graph  $R$  has a matching of size  $c$  is upper bounded by  $M \cdot (t^2/n)^c < (\ell^2 t^2/n)^c$ , because we consider  $M \cdot (t^2)^c$  events, and each event occurs with probability  $(1/n)^c$ . Specifically, each event corresponds to a choice of  $c$  disjoint pairs of seeds and a choice of a pair of samples per each pair of seeds, and this sequence of  $2c$  samples is uniformly distributed in  $[n]^{2c}$ . (Here we use the fact that the seeds generated by  $G'$  are uniformly distributed in an  $2c$ -wise independent manner, and that each sample generated by  $G$  is uniformly distributed in  $[n]$ .)

Recalling that  $t = \tilde{O}(n^{1/3})$  and  $\ell = O(\log n)$  (and  $\gamma = 1/\text{poly}(n)$ ), and that  $c$  is a constant chosen at our discretion, we conclude that with probability  $1 - \gamma$  the graph contains an independent set of size  $\ell - 2c$ , which means that the  $\ell$  seeds generate at least  $\ell - 2c$  disjoint samples. The final sampler outputs the union of  $\ell - 2c$  disjoint samples, and an arbitrary sequence of  $(\ell - 2c) \cdot t$  elements otherwise (i.e., if such a collection of disjoint samples does not exist). By the foregoing analysis this final sampler also satisfies the average sampling property of Eq. (1).  $\blacktriangleleft$

### 3.3 Applications to explicit constructions of extractors

We first describe the implication of the new averaging sampler on randomness extraction in  $\mathcal{AC}^0$  (from  $(n, k)$ -sources). We start by spelling out the construction obtained by instantiating Corollary 2.8 with the averaging sampler of Theorem 3.2.

► **Corollary 3.5** (using the averaging sampler of Theorem 3.2). *Let  $\beta \in (0, 1)$  be a constant,  $\delta > n^{-1/3}$  and  $t \in [\Theta(\delta^{-1} \log^2 n), \tilde{O}(n^{1/3})]$ . Suppose that  $E_0 : \{0, 1\}^t \times \{0, 1\}^{r_0} \rightarrow \{0, 1\}^m$  is a  $(\beta\delta \cdot t, \epsilon)$ -extractor that is computable by (uniform) constant-depth circuits of  $\text{poly}(n)$ -size. Then, there exists a  $(\delta \cdot n, 3 \cdot \max(\epsilon, \text{poly}(1/n)))$ -extractor  $E : \{0, 1\}^n \times \{0, 1\}^{r_0 + O(\log n)} \rightarrow \{0, 1\}^m$  that is computable by (uniform)  $\mathcal{AC}^0$  circuits. Furthermore, the depth of the circuits for  $E$  is only one unit more than the depth of the circuits for  $E_0$ , and if  $E_0$  is strong then so is  $E$ .*

(The constant  $1/3$  can be replaced by any constant smaller than  $1/2$ .)

**Proof.** Towards invoking Corollary 2.8, we set  $\mu' = \beta\delta/O(\log(1/\delta))$  and  $\mu = \frac{1+2\beta}{3\beta} \cdot \mu'$ . We use the  $(\mu, \mu', 1/\text{poly}(n))$ -averaging sampler  $S : \{0, 1\}^{O(\log n)} \rightarrow [n]^t$  of Theorem 3.2, while noting that  $\mu'/\mu = 3\beta/(1+2\beta)$  is a constant smaller than 1, whereas  $\mu > n^{-1/3}/\text{poly}(\log n)$  and  $t = \Omega(\delta^{-1} \log^2 n) = \Omega(\mu^{-1} \log n)$ . Invoking Corollary 2.8, while noting that  $\delta = \omega((\log n)/n)$  and that  $S$  is computable by uniform DNFs (resp., CNFs) of  $\text{poly}(n)$ -size, we obtain the desired extractor. ◀

Next, combining Corollary 3.5 with known extractors, which can be computed by constant-depth circuits of  $\text{poly}(n)$ -size, we derive the following –

► **Corollary 3.6** (extraction in  $\mathcal{AC}^0$  with seed of logarithmic length).

1. (Theorem 3.1, restated): *For every  $k(n) \geq n/\text{poly}(\log n)$  and  $\epsilon(n) = 1/\text{poly}(n)$ , there exist explicit  $\mathcal{AC}^0$  circuits that compute a strong  $(k(n), \epsilon(n))$ -extractor  $E : \{0, 1\}^n \times \{0, 1\}^{O(\log n)} \rightarrow \{0, 1\}^{\Theta(\log n)}$ . Furthermore, the circuits have depth  $4 + \left\lceil \frac{\log(n/k(n))}{\log \log n} \right\rceil$ .*
2. *For every  $k(n) \geq n/\text{poly}(\log n)$ ,  $m(n) = \text{poly}(\log n)$ , and  $\epsilon(n) = 1/\text{poly}(\log n)$ , there exist explicit  $\mathcal{AC}^0$  circuits that compute a strong  $(k(n), \epsilon(n))$ -extractor  $E : \{0, 1\}^n \times \{0, 1\}^{O(\log n)} \rightarrow \{0, 1\}^{m(n)}$ . Furthermore, the circuits have depth  $3 + \left\lceil \frac{\log(m(n) \cdot n/k(n))}{\log \log n} \right\rceil$ .*

The extractor of Part 2 has longer output (i.e.,  $m(n) = \text{poly}(\log n)$  rather than  $m(n) = O(\log n)$ ), but weaker quality than the extractor of Part 1 (i.e.,  $\epsilon(n) = 1/\text{poly}(\log n)$  rather than  $\epsilon(n) = 1/\text{poly}(n)$ ). We cannot afford the error-reduction procedures of [49], since these procedures seem sequential in nature. But it may be possible to implement the extractor of [31] (or another adequate extractor) by constant-depth circuits of sub-exponential size; in general, for a parameter  $\ell$  (to be set to  $O(\log n)$ ) and any  $\delta > 1/\text{poly}(\ell)$  and  $t = \text{poly}(\ell)$ , we merely need  $\exp(-\Omega(\ell))$ -error extractors for  $(\delta t, t)$ -sources that are implementable by constant-depth circuits of  $\exp(\ell)$ -size and extract  $(\delta t)^{\Omega(1)}$  bits.

**Proof.** Starting with Corollary 3.5, the two parts are proved by providing corresponding extractors. Specifically, we set  $\beta = 1/2$ ,  $\delta = k(n)/n$  and  $t = \Theta((\log n)^c \cdot \delta^{-1})$ , for a constant  $c \geq 1$  selected below, and pick an adequate extractor  $E_0 : \{0, 1\}^t \times \{0, 1\}^{r_0} \rightarrow \{0, 1\}^m$ .

For Part 1 we set  $c = 1$  and use the strong  $(\delta t/2, \epsilon)$ -extractor  $E_0 : \{0, 1\}^t \times \{0, 1\}^{r_0} \rightarrow \{0, 1\}^m$  asserted in [29, Sec. 5], which supports  $r_0 = O(m)$  for  $m = \Theta(\delta t) - \log(1/\epsilon)$ . Using  $\epsilon = 1/\text{poly}(n)$  and  $t = \Omega(\delta^{-1} \log n)$ , we have  $m = \Theta(\log n)$ , and all claims follow, since (as detailed next)  $E_0$  can be computed by (uniform)  $\text{poly}(n)$ -size circuits of depth  $2 + \lceil \log_m t \rceil = 3 + \lceil (\log(n/k(n)))/\log m \rceil$ .

To see that  $E_0$  can be computed by (uniform)  $\text{poly}(n)$ -size circuits of depth  $2 + \lceil \log_m t \rceil$ , we consider all  $\text{poly}(n)$  possible fixings of its seed, and observe that  $E_0$  can be computed by a depth-two circuit that selects the appropriate residual circuit. As noted in Section 3.1, viewing the  $t$ -bit source as a sequence of  $t/O(\log t)$  elements over a field  $F$  of size  $\text{poly}(t)$ , the residual computation (for each fixing of the seed to  $E_0$ ) amounts to a linear combination (over

$F$ ) of these field elements. Such linear combinations can be computed by depth  $1 + \lceil \log_m t \rceil$  circuits of  $\text{poly}(n)$ -size.

Turning to Part 2, we set  $c = \frac{\log m}{\log \log n} + 1$  (i.e.,  $m = (\log n)^{c-1}$ ) and use the strong  $(\delta t/2, \epsilon)$ -extractor  $E_0 : \{0, 1\}^t \times \{0, 1\}^{r_0} \rightarrow \{0, 1\}^m$  asserted by Trevisan [54], which supports  $r_0 = O(\log(t/\epsilon))$  and  $m = (\delta t/2)^{(c-1)/c}$ , provided that  $\delta t/2 > t^{\Omega(1)}$  and  $\epsilon = 1/\text{poly}(t)$ . Recalling that the residual computations corresponding to Trevisan's extractor amounts to a linear combination of the  $t$  bits, we conclude that these linear combinations can be computed by depth  $1 + \lceil \log_{\log n} t \rceil = 1 + \lceil c + \log_{\log n}(n/k(n)) \rceil$  circuits of  $\text{poly}(n)$ -size. ◀

### Application to explicit constructions of local extractors

Combining Theorem 3.2 with [55, Thm. 6.3] (see Corollary 2.8), yields improved parameters for explicit local extractors. Further improvement is obtained by using better extractors (i.e., the current state-of-art extractors of Guruswami *et al.* [31, Thm. 1.5]) than those available to Vadhan [55].<sup>10</sup>

► **Corollary 3.7** (implications to locally computable extractors). *For every constant  $\alpha \in (0, 1)$ ,  $k = \Omega(\text{poly}(\log n))$  and  $\epsilon = 1/\text{poly}(n)$ , and every  $t \geq \min(\Theta((n/k) \log^2 n), n)$ , there exists an explicit construction of a  $t$ -local  $(k, \epsilon)$ -extractor  $E : \{0, 1\}^n \times \{0, 1\}^{O(\log n)} \rightarrow \{0, 1\}^{\alpha \cdot (k/n) \cdot t}$ .*

**Proof Sketch.** Assume that  $t < n$ , since otherwise the claim is trivial (using  $t = n$  and [31, Thm. 1.5]). Now, setting  $\delta = k(n)/n$ ,  $\mu = \delta/O(\log(1/\delta))$  and  $t \geq \Theta(\mu^{-1} \log n)$ , combine the  $(\mu, \alpha\mu, 0.3\epsilon)$ -averaging sampler of Theorem 3.2 (which maps  $\{0, 1\}^{O(\log n)}$  to  $[n]^t$ ) with the  $(\alpha\delta t, 0.3\epsilon)$ -extractor of Guruswami *et al.* [31, Thm. 1.5] (set to map  $\{0, 1\}^t \times \{0, 1\}^{O(\log(t/\epsilon))}$  to  $\{0, 1\}^{\alpha^2 \delta t}$ ), where the combination is via Corollary 2.8. This yields extraction of  $\alpha^2 \delta t$  bits. ◀

## 3.4 An alternative averaging sampler

The drawback of the averaging sampler presented in Theorem 3.2 is that it does not apply to the case that both  $\mu < n^{-1/2}$  and  $\gamma < 1/n^2$  (or so). This was good enough for the applications to extraction in  $\mathcal{AC}^0$  presented in Section 3.3, but is not sufficient for other applications (see, e.g., Section 6, and specifically Corollary 6.2).

Recall that the reason for the lower bound on  $\mu$  is that we needed  $t^2 > \mu^{-2}$  to be (significantly) smaller than  $n$ , so that we may expect two random  $t$ -subsets of  $[n]$  to be disjoint. The latter condition was used for establishing the claim that the sampler always generate distinct elements (as required in the furthermore clause of Definition 2.6). We now achieve the latter goal while using a seed of length  $O(\log n)^2$ , rather than  $O(\log n)$ , which suffices for our application (i.e., Corollary 6.2).

► **Theorem 3.8** (an averaging sampler for all densities). *Let  $\alpha \in (0, 1)$  be an arbitrary constant. Then, for every  $\mu > 1/n$  and  $\gamma \geq 1/\text{poly}(n)$ , there exist explicit constant-depth  $\text{poly}(n)$ -size circuits  $C_n : \{0, 1\}^{O(\log n)^2} \rightarrow [n]^t$  that compute a  $(\mu(n), \alpha\mu(n), \gamma(n))$ -averaging sampler for any  $t \in [\Theta(\mu^{-1} \log^2 n), 0.1n]$ .*

**Proof Sketch.** Our starting point is the simple (pairwise independence) averaging sampler  $G$  asserted in Claim 3.3. Setting  $\ell = O(\log n)$ , we consider  $\ell$  independent invocations of

<sup>10</sup>The latter improvement is in the output length and in some cases in the deviation parameter  $\epsilon$ . Specifically, using only extractors that were available to Vadhan (i.e., [54, 49]), one can extract  $\Omega((k/n) \cdot t)^{0.999}$  bits (rather than  $\Omega((k/n) \cdot t)$  bits) for  $\epsilon = \max(\text{poly}(1/n), \exp(-t^{1-o(1)}))$ .

$G$  and the sequence of  $\ell \cdot t$  elements generated. Denoting this sequence (or multi-set) by  $S$ , note that each element occurs in  $S$  at most  $\ell$  times, since the sets generated by  $G$  have no repetitions. Denoting the number of occurrences of  $i$  (in the multi-set  $S$ ) by  $q_i$ , we use an  $\ell$ -wise independent sequence of length  $n$  over  $[\ell]$  in order to select each  $i \in [n]$  with probability  $q_i/\ell$ , resulting in a set  $S'$  (with no repetitions).

Using an  $\ell^{\text{th}}$  moment inequality (see Appendix A.2) it can be shown that the probability that the number of sampled elements is  $(1 \pm \epsilon) \cdot t$  is greater than  $1 - (\epsilon^{-2}\ell^2/t)^{\ell/2}$ , since the expected number is  $\ell \cdot t/\ell$  (whereas  $t \geq 2\epsilon^{-2}\ell^2 = O(\log^2 n)$ , since we refer to any constant  $\epsilon > 0$ ). Fixing any  $f : [n] \rightarrow [0, 1]$  such that  $\rho(f) \geq \mu$ , and assuming that  $\sum_{i \in S} f(i) \geq \alpha\mu \cdot \ell t$  (which occurs with probability at least  $1 - \gamma$  (cf. the proof of Theorem 3.2)), we apply the same reasoning to the  $f$ -value of the random set  $S'$ . Specifically, since  $t = \Omega(\mu^{-1} \log^2 n)$ , it follows that for any constant  $\epsilon > 0$  the probability that  $\sum_{i \in S'} f(i) < (1 - \epsilon) \cdot \alpha\mu \cdot t$ , is at most  $(\epsilon^{-2}\ell^2/\alpha\mu t)^{\ell/2} < \gamma$  (where here we use  $t \geq 2\epsilon^{-2}\alpha^{-1}\mu^{-1}\ell^2 = O(\mu^{-1} \log^2 n)$ ). Finally, we augment the set  $S'$  so to obtain a set of size exactly  $(1 + 2\epsilon) \cdot t$ .

It is left to show that the foregoing sampler can be implemented by uniform constant-depth circuits of  $\text{poly}(n)$ -size. This is not straightforward. Hence, the foregoing description is merely an intuitive motivation and the actual implementation proceeds as follows.

1. Obtaining the multi-set  $S$  (by using  $\ell$  invocations of  $G$ ), and generating an  $\ell$ -wise independent sequence  $(p_1, \dots, p_n) \in [\ell]^n$ .

The implementation of this step by uniform constant-depth circuits of size  $\text{poly}(n)$  is straightforward for  $G$  and quite standard for the  $\ell$ -wise independent sequence. The construction of an  $\ell$ -wise independent sequence over  $[\ell]$  is actually performed over  $\text{GF}(2^{\log n})$ , and involves taking various linear combinations of  $\ell$  field elements, which constitute the seed of the  $\ell$ -wise independence generator.

2. Computing the number of occurrences of each element (in  $S$ ) and producing the set  $S'$ . Specifically, the element  $e \in [n]$  is included in  $S'$  if and only if it occurs in  $S$  at least  $p_e$  times (i.e.,  $q_e \geq p_e$ ), where  $(p_1, \dots, p_n)$  is the sequence produced in Step 1.

This can be done by uniform constant-depth circuits of size  $\text{poly}(n)$  because the number of occurrences of element  $e \in [n]$  in  $S$  equals the cardinality of the maximal set  $I \subseteq [\ell]$  such that each index in  $I$  corresponds to an invocation of  $G$  that produced a sample that contains  $e$ . Specifically, the condition to check for each set  $I$  is that *for every  $i \in I$  there exists  $j \in [t]$  such that  $r_i + \phi_j s_i = e$ , where  $(r_i, s_i)$  is the seed used in the  $i^{\text{th}}$  invocation of  $G$ .* (Alternatively, we can count the number of occurrences of each  $e \in [n]$  in  $S$  by using a counter for small values, as in Step 3(b).) Indeed, we obtain a representation of  $S$  as an  $n$ -long sequence over  $\{0, 1, \dots, \ell\}$ , and produce (or sieve out) the set  $S'$  using the same representation, which means that  $S'$  is represented as an  $n$ -bit long sequence.

The above actions can be performed by uniform constant-depth circuits of size  $2^\ell \cdot \text{poly}(n) = \text{poly}(n)$ . Note, however, that the set  $S'$  is not in the standard format that we use throughout this paper, which is a format that seems essential for having  $\mathcal{AC}^0$  circuits compute a mapping of the form  $(x, S') \mapsto x_{S'}$ . The next steps are aimed at putting  $S'$  in the standard format.

3. Making progress towards obtaining a standard representation of  $S'$  (as a sequence of elements of  $[n]$ ) is done as follows.

- a. First, we select an  $\ell$ -wise independent hash function  $h : [n] \rightarrow [t/\ell^3]$  and hash  $S' \subset [n]$  to  $[t/\ell^3]$  such that, with probability at least  $1 - \gamma$ , each image  $i \in [t/\ell^3]$  is assigned  $(1 \pm 2\epsilon) \cdot \ell^3$  elements of  $S'$ . (The probabilistic claim holds by virtue of the  $\ell$ -wise independent hash function we use, while noting that  $(1 - \epsilon) \cdot \ell^3 > 2\epsilon^{-2} \cdot \ell^2$ .)

- b. Next, we rank the elements of  $S'$  that are hashed by  $h$  to each image  $i \in [t/\ell^3]$ .

This is done by using a (uniform  $\mathcal{AC}^0$ ) counter for small values (i.e., a uniform  $\mathcal{AC}^0$  circuits that counts the number of ones a string of length  $n$ , when guaranteed that this number does not exceed  $\text{poly}(\log n)$ ).<sup>11</sup> An explicit construction of such a counter was presented in [48], improving over [3, Sec. 5] (and subsequent works).<sup>12</sup>

Here, we count for each  $e \in (S' \cap h^{-1}(i))$  the number of elements in  $S'$  that are smaller than  $e$  and are hashed by  $h$  to  $h(e)$  (i.e., we compute  $\sum_{e' \in [n]} \chi_{e'}$ , where  $\chi_{e'} = 1$  if  $e' \in S'$  and  $h(e') = h(e)$ ).

At the end of the current step we obtain a representation of  $S'$  as a sequence of  $t/\ell^3$  (sorted) sets that are each of size  $(1 \pm 2\epsilon) \cdot \ell^3$ , where the latter claim holds with probability  $1 - 2\gamma$ . This is closed to the desired format, but is not quite there.

Note that the current step can be performed by uniform constant-depth circuits of size  $\text{poly}(n)$ , where the key observation is that such circuits can rank  $\text{poly}(\log n)$  many elements that reside in an array of length  $n$ . (The implementation of that  $\ell$ -wise independent hash functions from  $[n]$  to  $[n/\ell^3]$  is similar to the implementation of  $\ell$ -wise independent sequences over  $[\ell]$  discussed in Step 1.)

4. Obtaining a standard representation of the augmented set  $S'$  (as a sequence of elements of  $[n]$ ) is done as follows. Indeed, obtaining the standard representation is linked to augmenting the set  $S'$  to a set of size exactly  $(1 + 2\epsilon) \cdot t$ .

Using an  $\ell$ -wise independent sequence of length  $t$  over  $[n]$ , we first select  $t$  elements of  $[n]$ , while noting that (with probability at least  $1 - \gamma$ ) we obtain at least  $t/2$  distinct elements that are not in  $S'$ . (Here we use  $t \leq 0.1n$ .) Furthermore, the number of additional elements mapped to each image of the ( $\ell$ -wise independent) hash function  $h$  (used in Step 3) is at most  $(1 + \epsilon) \cdot \ell^3$  and the number of additional elements not in  $S'$  that are mapped to this image under  $h$  is at least  $\ell^3/2$ .

Denoting the multi-set of additional elements by  $A$  and fixing a good hash function  $h$ , note that for every  $i \in [t/\ell^3]$  it holds that (i)  $|S' \cap h^{-1}(i)| = (1 \pm 2\epsilon) \cdot \ell^3$ , (ii)  $|A \cap h^{-1}(i)| \leq (1 + \epsilon) \cdot \ell^3$ , and (iii)  $|(A \setminus S') \cap h^{-1}(i)| \geq \ell^3/2$ . Now, we rank the  $O(\ell^3)$  elements of  $A$  that are mapped by  $h$  to each image  $i \in [t/\ell^3]$ , just as we did with  $S'$  in Step 3, and use the two rankings in order to obtain a list of  $(1 + 2\epsilon) \cdot \ell^3$  elements that are mapped (by  $h$ ) to  $i$  such that this list contains all elements in  $S' \cap h^{-1}(i)$ . (Specifically, we place the  $j^{\text{th}}$  elements of  $S' \cap h^{-1}(i)$  in position  $j$ , and the  $j^{\text{th}}$  element of  $A \cap h^{-1}(i)$  in position  $|S' \cap h^{-1}(i)| + j \leq (1 + 2\epsilon)\ell^3$ .)<sup>13</sup> At this point we have the desired format.

If any of these steps failed, which happened with probability  $O(\gamma)$ , then we just output a fixed sequence.  $\blacktriangleleft$

► **Corollary 3.9** (using the averaging sampler of Theorem 3.8). *Let  $\beta \in (0, 1)$  be a constant,  $\delta = \Omega(\log n)/n$  and  $t \in [\Theta(\delta^{-1} \log^3 n), 0.1n]$ . Suppose that  $E_0 : \{0, 1\}^t \times \{0, 1\}^{r_0} \rightarrow \{0, 1\}^m$  is a  $(\beta\delta \cdot t, \epsilon)$ -extractor that is computable by (uniform) constant-depth circuits of  $\text{poly}(n)$ -size. Then, there exists a  $(\delta \cdot n, \epsilon + \text{poly}(1/n))$ -extractor  $E : \{0, 1\}^n \times \{0, 1\}^{r_0 + O(\log n)^2} \rightarrow \{0, 1\}^m$  that is computable by (uniform)  $\mathcal{AC}^0$  circuits.*

<sup>11</sup> These circuit can also detect the case that the guarantee is violated.

<sup>12</sup> An alternative construction is presented in Appendix A.3.

<sup>13</sup> Alternatively, we can just rank the the  $O(\ell^3)$  elements of  $S' \cup A$  that are mapped by  $h$  to each image  $i \in [t/\ell^3]$ , while considering the elements of  $S'$  as smaller than those of  $A$ . Either way, the fact that  $A$  is a multi-set is irrelevant to our analysis, and the procedure just uses each element of  $A$  as if it has appeared with multiplicity 1. In contrast, one should not ignore multiplicity when considering the multi-set  $S$ , since multiplicity may have an effect on the sampling properties of  $S$ .

**Proof Sketch.** As in the proof of Corollary 3.5, towards invoking Corollary 2.8, we set  $\mu' = \beta\delta/O(\log(1/\delta))$  and  $\mu = \frac{1+2\beta}{3\beta} \cdot \mu'$ . But here we use the  $(\mu, \mu', 1/\text{poly}(n))$ -averaging sampler  $S : \{0, 1\}^{O(\log n)^2} \rightarrow [n]^t$  of Theorem 3.8, while noting that  $\delta = \Omega((\log n)/n)$  and  $t = \Omega(\mu^{-1} \log^2 n)$ . Invoking the alternative part of Corollary 2.8, while noting that  $\delta = \Omega((\log n)/n)$ , the claim follows.  $\blacktriangleleft$

## 4 Extraction from Block Sources

We consider block sources as defined by Chor and Goldreich [17], and not their generalization as used by Zuckerman [61, 62] (see also [46]) and subsequent works.

► **Definition 4.1** (block sources [17]<sup>14</sup>). An  $(n, (\ell, b))$ -block source is a sequence of (possibly dependent) random variables  $X = (X_1, \dots, X_n) \in \{0, 1\}^{n\ell}$  such that for every  $i \in [n]$  and  $x_1, \dots, x_i \in \{0, 1\}^\ell$  it holds that

$$\Pr[X_i = x_i | X_1 \circ \dots \circ X_{i-1} = x_1 \circ \dots \circ x_{i-1}] \leq 2^{-b}$$

It follows that  $\Pr[X_1 \circ \dots \circ X_n = x_1 \circ \dots \circ x_n] \leq (2^{-b})^n$ , which means that  $X$  is an  $(n\ell, nb)$ -source.

### A short discussion

Zuckerman and subsequent works considered a generalized definition of block sources with varying block-length, where typically the blocks lengths decrease drastically (e.g.,  $|X_i| < |X_{i-1}|/2$ ). This was used as a methodological step towards constructing extractors for general min-entropy sources. While the study of randomness extractors for general min-entropy sources proved to have numerous applications (most of which were not envisioned originally), we believe that the original motivation of *extracting high-quality randomness out of low-quality sources of randomness* is extremely important. Furthermore, we believe that block sources are a very realistic model of poor sources of randomness, and hence we believe that extracting high-quality randomness from such sources is of great importance.

Of course, one can extract randomness from block sources by using a corresponding extractor for general min-entropy sources, since any  $\epsilon$ -error extractor for  $(n\ell, nb)$ -sources is an  $\epsilon$ -error extractor for  $(n, (\ell, b))$ -sources. Yet, more advantageous constructions may be possible for block sources, because the latter are a very restricted special case. In particular, it may be desirable to extract randomness on-the-fly (i.e., in a block-by-block manner), rather than wait for the entire source outcome, and it may be desirable to do so without storing much information. (Of course, this is impossible for general min-entropy sources.)

### 4.1 A simple extractor

The following construction extract randomness separately from each block, using the same random seed, and without sharing any other information among the (block-based) steps of the extraction process. We stress that using the same seed for extraction from all the blocks harms the quality of the output in a small and minimal manner.

<sup>14</sup> Actually, block sources were defined in [17] as being an infinite sequence of random variables that satisfy the (conditional) min-entropy bound.



► **Theorem 4.2** (a simple extractor for block sources). *Let  $E : \{0, 1\}^\ell \times \{0, 1\}^r \rightarrow \{0, 1\}^m$  be a strong  $(b, \epsilon)$ -extractor. Then,  $E' : \{0, 1\}^{n\ell} \times \{0, 1\}^r \rightarrow \{0, 1\}^{nm}$  defined by  $E'(x_1 \circ \dots \circ x_n, s) = E(x_1, s) \circ \dots \circ E(x_n, s)$  is a strong  $n \cdot \epsilon$ -error extractor for  $(n, (\ell, b))$ -block sources.*

Theorem 4.2 seems to be folklore. In particular, it follows as a special case of Lemma 5.7 of Guruswami *et al.* [31], but this fact is not transparent because their result refers to a “block chaining” construction; that is, in their construction, the original seed is used on the last block, and extraction from the  $i^{\text{th}}$  block (via  $E_i$ ) is used both for obtaining a part of the output and a seed for extraction from the  $i - 1^{\text{st}}$  block.<sup>15</sup> Given a strong extractor  $E$  as above, one should first define an auxiliary extractor  $E_1(x, s) = (s, E(x, s))$ , and then apply [31, Lem. 5.7] with  $E_i = E_1$  (for all  $i$ 's). (For sake of self-containment, we provide a direct and detailed proof of Theorem 4.2.)

**Proof.** We need to upper bound the statistical distance between  $U_{nm+r}$  and  $E(X_1, U_r) \circ \dots \circ E(X_n, U_r) \circ U_r$ , where all occurrences of  $U_r$  represent the same outcome and  $X = (X_1, \dots, X_n) \in \{0, 1\}^{n\ell}$  is an  $(n, (\ell, b))$ -block source. We prove the claim by induction on  $n$ , where the base case (of  $n = 1$ ) is immediate by the hypothesis regarding  $E$ . In the induction step we proceed as follows, using the notation  $X_{[j,k]} = (X_j, \dots, X_k)$  and  $E'(X_{[j,k]}, U_r) = E(X_j, U_r) \circ \dots \circ E(X_k, U_r)$ , where the choice of  $i \in [n - 1]$  is immaterial (i.e., any  $i \in [n - 1]$  will do):

$$\begin{aligned} & \Delta[E'(X, U_r) \circ U_r; U_{nm} \circ U_r] \\ &= \Delta[E'(X_{[1,i]}, U_r) \circ E'(X_{[i+1,n]}, U_r) \circ U_r; U_{im} \circ U_{(n-i)m} \circ U_r] \\ &\leq \Delta[E'(X_{[1,i]}, U_r) \circ E'(X_{[i+1,n]}, U_r) \circ U_r; E'(X_{[1,i]}, U_r) \circ U_{(n-i)m} \circ U_r] \\ &\quad + \Delta[E'(X_{[1,i]}, U_r) \circ U_{(n-i)m} \circ U_r; U_{im} \circ U_{(n-i)m} \circ U_r] \\ &\leq \Delta[X_{[1,i]} \circ E'(X_{[i+1,n]}, U_r) \circ U_r; X_{[1,i]} \circ U_{(n-i)m} \circ U_r] \tag{3} \\ &\quad + \Delta[E'(X_{[1,i]}, U_r) \circ U_r; U_{im} \circ U_r] \tag{4} \end{aligned}$$

where the last inequality uses the fact that  $\Delta[\Pi(Y); \Pi(Z)] \leq \Delta[Y; Z]$  holds for any random process  $\Pi$ . Using the induction hypothesis (regarding extraction from the  $(i, (\ell, b))$ -source  $X_{[1,i]}$ ), Eq. (4) is upper bounded by  $i \cdot \epsilon$ . So we turn to analyze Eq. (3). Letting  $X'_x$  denote the distribution of  $X_{[i+1,n]}$  conditioned on  $X_{[1,i]} = x$ , we get

$$\begin{aligned} & \Delta[X_{[1,i]} \circ E'(X_{[i+1,n]}, U_r) \circ U_r; X_{[1,i]} \circ U_{(n-i)m} \circ U_r] \\ &= \mathbf{E}_{x \leftarrow X_{[1,i]}} [\Delta[E'(X'_x, U_r) \circ U_r; U_{(n-i)m} \circ U_r]] \\ &\leq (n - i) \cdot \epsilon \end{aligned}$$

where the inequality uses the induction hypothesis regarding extraction from the  $(n - i, (\ell, b))$ -source  $X'_x$ . The claim follows. ◀

### Relevance to the study of extraction in $\mathcal{AC}^0$

Theorem 4.2 reduces the complexity of extraction from  $(n, (\ell, b))$ -block sources to the complexity of extraction from  $(\ell, b)$ -sources. In particular, we get –

<sup>15</sup>Specifically, in [31, Lem. 5.7],  $E'(x_1 \circ \dots \circ x_n, s) = y_1 \circ \dots \circ y_n$  (or rather  $E'(x_1 \circ \dots \circ x_n, s) = s_0 \circ y_1 \circ \dots \circ y_n$ ), where  $s_n = s$  and  $(s_{i-1}, y_i) \leftarrow E_i(x_i, s_i)$  for  $i = n, \dots, 1$ .



► **Corollary 4.3** ( $\mathcal{AC}^0$  extractors for block sources). *For every  $\ell(n) = \text{poly}(\log n)$  and  $\epsilon(n) = 1/\text{poly}(n)$  there exist  $b(n) = O(\log n)$  and explicit  $\mathcal{AC}^0$  circuits that compute a strong  $\epsilon(n)$ -error extractor  $E : \{0, 1\}^{n \cdot \ell} \times \{0, 1\}^{O(\log n)} \rightarrow \{0, 1\}^{n \cdot \Omega(\log n)}$  for  $(n, (\ell, b))$ -block sources. Furthermore, the circuits have depth  $4 + \frac{\log(\ell(n)/b(n))}{\log \log n}$ .*

Note that  $b(n)/\ell(n)$  is the min-entropy rate of the source.

**Proof.** Wishing to use the construction of Theorem 4.2, we (again) use the  $(b, \epsilon)$ -extractor  $E : \{0, 1\}^{\ell(n)} \times \{0, 1\}^{O(\log n)} \rightarrow \{0, 1\}^{\Omega(\log n)}$  asserted in [29, Sec. 5]. As noted in the proof of Theorem 3.1, this extractor can be computed by (uniform)  $\text{poly}(n)$ -size circuits of depth  $3 + \log_{m(n)} \ell(n) = 4 + (\log(\ell(n)/b(n)))/\log \log n$ . ◀

## 4.2 On converting min-entropy sources into block sources

In light of the relative ease of extracting randomness from block sources, it is natural to try to convert general min-entropy sources into block sources. Such conversion is intended to be easier than extraction; specifically, it amounts to sampling bits of the original source and arranging them in blocks.

This idea goes back to Zuckerman’s work [61, 62], but in all known incarnations each block is sampled using fresh randomness. This is typically not a problem when the number of blocks is small, but in our context we wish the number of blocks to be large. The question addressed here is *whether conversion is possible using randomness complexity that is significantly smaller than the number of desired blocks*.

Following is a definition that captures what we mean by conversion, which we call *blocking* (i.e., converting into a block source). Loosely speaking, a blocker is given a general  $(n, \delta n)$ -source, and needs to “produce” a block source with  $m > 1$  blocks such that each blocks has min-entropy rate at least  $\delta'$  (conditioned on prior blocks). Of course, we consider  $\delta' \in (0, \delta)$  and blocks of length  $s \leq n/m$ . In terms of the following definition of blockers, we ask whether we may have  $r = o(m)$ .

► **Definition 4.4** (converting general min-entropy sources to block sources). For  $n, m, s, r \in \mathbb{N}$  and  $\epsilon, \delta, \delta' \in [0, 1]$ , consider a generator  $S : \{0, 1\}^r \rightarrow ([n]^s)^m$ , which on input a seed  $u \in \{0, 1\}^r$  outputs an  $m$ -long sequence of  $s$ -subsets of  $[n]$ , denoted  $(S_1(u), \dots, S_m(u))$ . We say that  $S$  is a  $(\delta, \delta', \epsilon)$ -blocker if for every  $(n, \delta n)$ -source  $X$ , for at least  $1 - \epsilon$  of the choices of  $u \in \{0, 1\}^r$ , it holds that  $(X_{S_1(u)}, \dots, X_{S_m(u)})$  is an  $(m, (s, \delta' s))$ -block source.

Definition 4.4 is analogous to the strong version of extraction (i.e., strong extractors). A milder requirement is that  $(X_{S_1(u)}, \dots, X_{S_m(u)})$  is  $\epsilon$ -close to a  $(m, (s, \delta' s))$ -block source. For simplicity, we consider the stronger notion first and postpone the consideration of the milder notion to later (see Remarks 4.2.1 and 4.2.2).

While we do not resolve the foregoing question, we present two illustrations for its difficulty. First, we show that the requirements from a blocker must be more demanding than the requirements from a generator of a sequence of disjoint sets in which each set in the sequence has good sampling properties. This is shown for the case of  $m = 2$ , albeit using a somewhat contrived construction. Next, we show that a natural construction fails too; this is shown for  $m = n^{\Omega(1)}$  and  $r = O(\log n)$ . In both cases, the reader may think of small but constant values of  $\delta > \delta' > 0$  and of  $s = \text{poly}(\log n)$ . (Recall that randomness extraction from block sources can be performed in  $\mathcal{AC}^0$  when the block length is poly-logarithmic.)

### 4.2.1 1st illustration: Sampling does not suffice (even for two blocks)

Here we show that a “two-set sampler” is not necessarily a good blocker. We shall show this by considering a very general and natural definition of a two-set sampler, and show that there exists such constructs that fails as blockers.

We shall refer to a rather generic notion of a two-set sampler that generates pairs of sets such that some property  $\mathcal{P}$  is satisfied for each individual set. The property  $\mathcal{P}$  is actually a collection of conditions (or properties), and it is required that each condition is satisfied (with specified probability). Of course, it is also required that the two generated sets are disjoint.

► **Definition 4.5** (two-set sampler w.r.t a class of properties  $\{\mathcal{P}_i\}$ ). For  $n, s, r \in \mathbb{N}$ , let  $S : \{0, 1\}^r \rightarrow [n]^s \times [n]^s$  and denote  $(S_1(u), S_2(u)) = S(u)$ . For  $\epsilon \in [0, 1]$  and  $\mathcal{P}_i \subseteq [n]^s$  for every  $i \in [t]$ , we say that  $S$  is a two-set sampler w.r.t  $(\epsilon, \{\mathcal{P}_i : i \in [t]\})$  if the following two conditions hold:

1. For each  $\sigma \in \{1, 2\}$  and  $i \in [t]$ , it holds that  $\Pr[S_\sigma(U_r) \in \mathcal{P}_i] \geq 1 - \epsilon$ .
2. For every  $u \in \{0, 1\}^r$ , the sets  $S_1(u)$  and  $S_2(u)$  are disjoint.

A natural class of properties are those that correspond to a  $(\delta, \epsilon)$ -averaging sampler. In this case, the properties correspond to subsets of  $[n]$  and the property corresponding to a set  $T \subseteq [n]$  consists of all  $s$ -subsets having  $(\rho(T) \pm \delta) \cdot s$  elements in  $T$ , where  $\rho(T)$  is the density of  $T$  in  $[n]$ .

We could have stated the result only for averaging samplers (i.e., for the aforementioned properties that correspond to them), but we believe that it good to state it in greater generality. Since  $\{\mathcal{P}_i\}$  is totally generic, it is not clear that two-set samplers w.r.t it exist. For this reason, the following theorem assumes the existence of a two-set sampler w.r.t  $\{\mathcal{P}_i\}$ . Furthermore, we also require that  $\{\mathcal{P}_i\}$  is closed under relabeling of  $[n]$ ; that is, for every permutation  $\pi : [n] \rightarrow [n]$  and every  $i$  there exists a  $j$  such that  $\mathcal{P}_j = \{\pi(A) : A \in \mathcal{P}_i\}$ . Note that the properties corresponding to averaging samplers are indeed closed under relabeling of  $[n]$ . The following theorem says that whenever two-set samplers (w.r.t  $\mathcal{P}$ ) exist at all, there exists such samplers that fail as blockers (for the case of  $m = 2$ ).

► **Theorem 4.6** (two-set samplers are not necessarily blockers). *Let  $n, s \in \mathbb{N}$  such that  $n = \omega(s^2)$ ,  $\epsilon \in [0, 1]$ , and  $\mathcal{P} = \{\mathcal{P}_i \subseteq [n]^s\}$  be a collection of properties that is closed under relabeling of  $[n]$ . If there exists a two-set sampler w.r.t  $(\epsilon, \mathcal{P})$ , then there exists a two-set sampler w.r.t  $(2\epsilon, \mathcal{P})$  that is not a  $(0.5, o(1), 1 - o(1))$ -blocker. Furthermore, there exists an  $(n, 0.5n)$ -source  $X$  such that given  $X$ , with probability at least  $1 - 2s^2/n$ , this two-set sampler outputs a source of two blocks such that the second block is totally determined by the first block.*

**Proof.** We start with an arbitrary two-set sampler w.r.t  $(\epsilon, \mathcal{P})$ , denoted  $S : \{0, 1\}^r \rightarrow [n]^s \times [n]^s$ . Consider a random matching  $\pi : [n] \rightarrow [n]$  (i.e., a random bijection  $\pi$  such that  $\pi(i) \neq i$  and  $\pi(\pi(i)) = i$  for every  $i \in [n]$ ), and note that, w.v.h.p, for an  $1 - O(s^2/n)$  fraction of the  $u$ 's it holds that  $\pi(S_1(u)) \cap S_1(u) = \emptyset$ . Let us call such  $u$ 's good for  $\pi$ , and note that there exists a matching  $\pi$  for which the fraction of good  $u$ 's is at least  $1 - O(s^2/n) = 1 - o(1)$ . Fix such a  $\pi$  and define a new two-set sampler  $S'$ , which uses a seed  $(u, u') \in \{0, 1\}^{2r}$ , as follows:

1. If  $\pi(S_1(u)) \cap S_1(u) = \emptyset$ , let  $S'(u, u') = (S_1(u), \pi(S_1(u)))$  and call  $u$  good;
2. Otherwise (i.e.,  $\pi(S_1(u)) \cap S_1(u) \neq \emptyset$ ) let  $S'(u, u') = S(u')$ .

Note that  $1 - o(1)$  of the  $u$ 's are good, and that  $S'$  is a two-set sampler w.r.t  $(2\epsilon, \mathcal{P})$ . To prove the latter assertion, let us first consider  $S'_1(U_{2r})$ , and denote the set of good  $r$ -bit

strings by  $G$ . Then, for every  $\mathcal{P}_i$ ,

$$\begin{aligned} \Pr[S'_1(U_{2r}) \notin \mathcal{P}_i] &= \Pr_{u \leftarrow U_r}[S_1(u) \notin \mathcal{P}_i \wedge u \in G] + \Pr_{(u,u') \leftarrow U_{2r}}[S_1(u') \notin \mathcal{P}_i \wedge u \notin G] \\ &< \Pr_{u \leftarrow U_r}[S_1(u) \notin \mathcal{P}_i] + \Pr_{(u,u') \leftarrow U_{2r}}[S_1(u') \notin \mathcal{P}_i]. \end{aligned}$$

The same analysis applies to the second set (i.e.,  $S'_2(U_{2r})$ ), except that here  $S_1(u)$  is replaced by  $S_1(\pi(u))$  and the “closure under relabeling” hypothesis is used. Specifically, suppose that  $\mathcal{P}_j = \{\pi^{-1}(A) : A \in \mathcal{P}_i\}$ , then:

$$\begin{aligned} \Pr[S'_2(U_{2r}) \notin \mathcal{P}_i] &= \Pr_{u \leftarrow U_r}[S_1(\pi(u)) \notin \mathcal{P}_i \wedge u \in G] + \Pr_{(u,u') \leftarrow U_{2r}}[S_2(u') \notin \mathcal{P}_i \wedge u \notin G] \\ &< \Pr_{u \leftarrow U_r}[S_1(\pi(u)) \notin \mathcal{P}_i] + \Pr_{(u,u') \leftarrow U_{2r}}[S_2(u') \notin \mathcal{P}_i] \\ &= \Pr_{u \leftarrow U_r}[S_1(u) \notin \mathcal{P}_j] + \Pr_{u' \leftarrow U_r}[S_2(u') \notin \mathcal{P}_i]. \end{aligned}$$

Hence,  $S'$  is a two-set sampler w.r.t  $(2\epsilon, \mathcal{P})$ . In contrast to this fact, as shown next, it turns out that  $S'$  fails miserably as a blocker.

Let  $X$  be uniform over the set of strings  $\{x : (\forall i \in [n]) x_i = x_{\pi(i)}\}$ , which has cardinality  $2^{n/2}$ . This means that  $X$  has min-entropy  $n/2$ . On the other hand, whenever  $u$  is good, we have that  $X_{S'_2(u)}$  is determined by  $X_{S'_1(u)}$ , since in this case  $S'_2(u) = \pi(S'_1(u))$ , which implies  $X_{S'_2(u)} = X_{\pi(S'_1(u))} = X_{S'_1(u)}$ . Hence, with probability at least  $1 - 2s^2/n = 1 - o(1)$  (i.e., whenever  $u$  is good), the second block in the source  $(X_{S'_1(u)}, X_{S'_2(u)})$  has min-entropy zero conditioned on the first block. ◀

► **Remark (non-strong blocking fails too).** Actually, the proof of Theorem 4.6 applies also to the weaker notion of blocking in which it is only required that  $(X_{S'_1(U_r)}, X_{S'_2(U_r)})$  is  $\epsilon$ -close to a  $(2, (s, \delta's))$ -block source. The proof implies that  $(X_{S'_1(U_r)}, X_{S'_2(U_r)})$  is  $2s^2/n$ -close to a source in which the second block equals the first block, and thus has no conditional entropy at all. It follows that this sampled source is far from any block source in which the second block has even just few bits of min-entropy.

## Digest

Note that Theorem 4.6 does not refer to the randomness complexity of the two-set sampler. In such a setting, we know that blockers (let alone for  $m = 2$ ) do exist. Hence what Theorem 4.6 asserts is only that requirements regarding the sampling features of individual sets generated by a two-set samplers do not imply that this two-set sampler is a blocker.

The reason that two-set samplers may fail as blockers is that their definition makes too mild requirements regarding the relation between the two generated sets. Indeed, Definition 4.4 only requires that these two sets be disjoint. The proof of Theorem 4.6 capitalizes on this fact and uses a fixed matching of elements between all pairs of sets (i.e., for a fixed matching  $\pi$ , the elements of the second set are typically the  $\pi$ -mates of the elements of the first set).

### 4.2.2 2nd illustration: A natural candidate that fails

The proof of Theorem 4.6 relies on a contrived example and shows that such an example exists no matter what “sampling property” (regarding individual sets) is considered. Here we take an opposite approach: We consider a natural construction (and do not specify the sampling properties that it satisfies). Specifically, we shall present a natural multi-set sampler, which we believe most readers may find a natural candidate for a good blocker, and show that it actually fails as a blocker. The multi-set sampler (and candidate blocker) refers to the case of  $m = n^{\Omega(1)}$ ,  $s = \text{poly}(\log n)$ , and  $r = O(\log n)$ . (Recall that these are

the parameters that we want in order to extract  $n^{\Omega(1)}$  bits in  $\mathcal{AC}^0$  via conversion to block sources.)

**A rhetoric question:** *What is more natural than trying the standard  $O(1)$ -wise independent generator?* Indeed, let us consider it.

► **Construction 4.7** (constant-wise independent multi-set sampler). *Let  $F$  be a finite field of size  $n$  and  $c \geq 2$  be a constant, and consider the  $c$ -wise independent generator, denoted  $G : F^c \rightarrow F^n$ , such that  $G(u) = (g_1(u), \dots, g_n(u))$  and  $g_j(u_1, \dots, u_c) = \sum_{i \in [c]} u_i \alpha_j^{i-1}$ , where  $\alpha_1, \dots, \alpha_n$  are distinct field elements. For  $m = n^{\Omega(1)}$  and  $s = \text{poly}(\log n)$ , consider the sampler  $S(u) = (S_1(u), \dots, S_m(u))$  such that  $S_i(u) = \{g_{(i-1)s+1}(u), \dots, g_{is}(u)\}$ .*

Note that  $S : F^c \rightarrow (F^s)^m$ , which means that this sampler has a seed of length  $c \log_2 n$ . *Is this sampler not a natural candidate for a blocker?* Well, as we show next, it fails badly.

- *Preliminaries:* We assume that for some  $d \in \mathbb{N}$  it holds that  $ms \in (0.9 \pm 0.1) \cdot n^{1/d}$  (or so).<sup>16</sup> Let  $F = K^d$ , where  $K$  is a finite field, and let  $H \subset K$  of sufficiently small constant density (say, density  $1/3cd$ ). Hence,  $k \stackrel{\text{def}}{=} |H^d| = \Omega(n)$ , since  $n = |K^d|$  and  $|H| = \Omega(|K|)$ .
- *A class of (affine) sources:* Next, for any  $f : H^d \rightarrow K$ , consider its low-degree extension  $f' : K^d \rightarrow K$ , and let  $X \in K^n$  be uniformly distributed among all  $d$ -variate polynomials of individual degree  $|H|$ ; that is, for a uniformly distributed  $f : H^d \rightarrow K$ , let  $X_\alpha = f'(\alpha)$  for every  $\alpha \in K^d \equiv [n]$ . So  $X$  has “min-entropy”  $k$  (in units of symbols in  $K$ ). Indeed, this is a source over the alphabet  $K$  (rather than over  $\{0, 1\}$ ).

By the way, it is an affine source, since the values of all  $X_\alpha$ 's (i.e., the polynomial  $f'$ ) are determined as linear combination of the values at  $\alpha \in H^d$  (i.e., the function  $f$ ).

- *The foregoing sampler  $S$  as a candidate blocker:* Now, for any  $u = (u_1, \dots, u_c) \in [n]^c \equiv (K^d)^c$ , the sampler  $S : (K^d)^c \rightarrow ((K^d)^s)^m$  produces the sampled source  $(X_{S_1(u)}, \dots, X_{S_m(u)})$ . Recall that  $ms \in (0.9 \pm 0.1) \cdot |K|$  and  $|H| = |K|/3cd$ .

Let  $X$  be a generic source from the above class. Plugging in the definition of  $S_i$ , note that the  $j^{\text{th}}$  element in the  $i^{\text{th}}$  block of the sampled source is  $X_{g_{(i-1)s+j}(u)} = X_{C_u(\alpha_{(i-1)s+j})}$ , where  $C_u(\alpha) = \sum_{\ell \in [c]} u_\ell \alpha^{\ell-1}$ . Now, suppose that  $\alpha_1, \dots, \alpha_{ms} \in F = K^d$  are all in the base field  $K$  (which is possible since  $ms < |K|$ ). Then, it suffices to define  $C_u$  on  $K$  (i.e.,  $C_u : K \rightarrow K^d$ ), which means that  $C_u$  is a  $(c-1)$ -degree curve over  $K^d$  (i.e., it is a curve over  $K^d$ , defined based on  $u \in (K^d)^c$ , with a free parameter in  $K$ ).

Recalling that  $X$  is defined in term of the low-degree extension  $f'$  of a random function  $f : H^d \rightarrow K$ , we have  $X_{C_u(\alpha_{(i-1)s+j})} = f'(C_u(\alpha_{(i-1)s+j}))$ , where  $f' \circ C_u : K \rightarrow K$  (via  $K^d$ ) is a degree  $(c-1)d|H|$  univariate polynomial over  $K$  (since  $f'$  is a  $d$ -variate polynomial of individual degree  $|H|$  and  $C_u$  is a curve of degree  $c-1$ ). Hence, for every seed  $u$  of the sampler, less than  $cd|H|$  symbols of the sampled source determine all other symbols (of the sampled source); in particular, *the first  $cd|H|/s$  blocks of the sampled source  $(X_{S_1(u)}, \dots, X_{S_m(u)})$  fully determine the remaining  $m - (cd|H|/s)$  blocks.* Recall that  $cd|H|/s = |K|/3s < m/2$  (by our choice of  $H$  and  $K$ ). ◀

► **Remark (non-strong blocking fails too).** Since there are only polynomially many curves  $C_u$ , using few additional symbols (*let alone few additional blocks*), we can determine which curve is used, and determine the remaining blocks based on this. Typically, each additional symbol

<sup>16</sup> After all, if the above sampler is a good blocker for  $m = n^{\Omega(1)}$  and  $s = \text{poly}(\log n)$ , then it should be a good blocker also when  $n$  is an integer power of  $ms$  (or so).

read from the sampled source, beyond the  $cd|H|$  symbols that suffice for extrapolation, cuts the number of possible curves by a factor of  $|K|$ .

### Critique

One may raise two reasonable objections to our example. (1) We considered sources over a large (non-binary) alphabet  $K$ ; (2) We made the sampler use the  $|K|$  elements of  $K \subset K^d$  as the coefficients in the  $c$ -wise independent sequence (of length  $|K|$ ). We believe that both objections are not acute.

Objection (1) can be addressed by encoding the elements of  $K$  by binary strings of length  $\ell = \log |K|$ . This requires an analogous modification of the sampler in which indices in  $[n]$  are replaced by  $\ell$ -sets of  $[n\ell]$  (i.e.,  $i$  is replaced by  $\{(i-1)\ell+1, \dots, i\ell\}$ ).<sup>17</sup> (Alternatively, one may argue that the question for arbitrary alphabets is as natural.) Objection (2) can be addressed by claiming that the construction should work regardless of the choice of field elements, let alone that the same argument holds when choosing field elements that reside on any line (or low degree curve) in  $K^d$  (instead of residing in  $K$ ). Indeed, we do not recall any application of  $c$ -wise independence that insists on a “non-structured” choice of the field elements (whenever there is a choice at all). Actually, it is quite natural to use a structured choice of field elements.

### 4.2.3 Discussion

Note that both counterexamples utilize affine sources, whereas it is possible to convert affine sources (having min-entropy at least  $k = n/\text{poly}(\log n)$ ) into block sources (using a seed of logarithmic length and obtaining blocks of length  $s = \text{poly}(n/k)$ ). Specifically, as hinted up-front, it is well known that conversion into  $m$  blocks is possible if one is willing to use a seed of length  $r = O(m \log n)$  (by repeated sampling with fresh random seeds). Since the number of affine sources is less than  $2^{n^2}$ , we may infer that there exists a set of  $O(n^2/\delta\epsilon)$  such that for an affine source of min-entropy  $\delta n$  at least a  $1 - \epsilon$  fraction of these seeds yield an  $(m, (s, 0.5\delta s))$ -block source.

The forgoing argument cannot be applied to general  $(n, \delta n)$ -sources, since their number is too large. But it is not inconceivable that a more refined counting argument may work. More generally, we ask –

► **Open Problem 4.8** (blockers of logarithmic randomness). *Does there exist a  $(\delta, \delta', o(1))$ -blockers  $S : \{0, 1\}^{O(\log n)} \rightarrow ([n]^s)^m$  for constant  $\delta' \in (0, \delta) \subset (0, 1)$  and  $m = \omega(1)$ ? What about  $m = n^{\Omega(1)}$  and  $s = \text{poly}(\log n)$ ?*

A positive answer to the latter question is a sufficient but possibly not necessary condition for a positive resolution of Problem 1.6. Of course, a negative resolution of Problem 1.6 would imply a negative answer to Problem 4.8.

## 5 Extraction from Block-Fixing Sources

In this section, we consider  $\mathcal{AC}^0$  extractors for bit-fixing sources, a model first considered by Chor *et al.* [18]. In this model the min-entropy bound  $k$  denotes the number of bits that are random in the source, whereas the other  $n - k$  bits are fixed *obliviously* of the values of the

<sup>17</sup>Indeed, this merely moves the problem from the alphabet to the sampler (which now samples indices by taking all indices in each sampled block), and one may object to viewing such a sampler as natural.

random bits.<sup>18</sup> Actually, we consider a generalization to block-fixing sources, first considered in [41].

► **Definition 5.1** (block-fixing sources [18, 41]). An  $(n, k, \ell)$ -block-fixing source is a sequence of random variables  $X = (X_1, \dots, X_n) \in \{0, 1\}^{n\ell}$  such that there exists a set of at least  $k$  indices  $I \subseteq [n]$  and a sequence  $(x_1, \dots, x_n) \in \{0, 1\}^{n\ell}$  such that  $X_I$  is uniformly distributed over  $\{0, 1\}^{|I|\ell}$  and  $X_i = x_i$  for every  $i \in [n] \setminus I$ . The special case of  $\ell = 1$  is referred to as a  $(n, k)$ -bit-fixing source; that is, a  $(n, k, 1)$ -block-fixing source is called a  $(n, k)$ -bit-fixing source.

Note that extractors for block-fixing sources need not preserve the block structure in their output, although the extractors presented in Theorem 5.2 do preserve this structure. In general, for every  $\ell$ , an  $\epsilon$ -error extractor for  $(n\ell, k\ell)$ -bit-fixing sources is an  $\epsilon$ -error extractor for  $(n, k, \ell)$ -block-fixing sources, and ditto for strong extractors. It is also easy to see that if  $E : \{0, 1\}^n \times \{0, 1\}^{r(n)} \rightarrow \{0, 1\}^{m(n)}$  is a strong  $\epsilon$ -error extractor for  $(n, k)$ -bit-fixing sources, then  $E' : \{0, 1\}^{\ell \cdot n} \times \{0, 1\}^{r(n)} \rightarrow \{0, 1\}^{\ell \cdot m(n)}$  given by

$$E'((x_{1,1}, \dots, x_{n,\ell}), s) = E((x_{1,1}, \dots, x_{n,1}), s) \circ \dots \circ E((x_{1,\ell}, \dots, x_{n,\ell}), s)$$

is a strong  $\ell\epsilon$ -error extractor for  $(n\ell, k\ell)$ -bit-fixing sources. (An analogous statement for ordinary extractors seems to require using  $\ell$  different seeds, and so the seed length becomes  $\ell \cdot r(n)$ .)

In Section 5.1, we present (strong) extractors for  $(n, n/\text{poly}(\log n), \ell)$ -block-fixing sources that use a seed of logarithmic length and extract  $n\ell/\text{poly}(\log n)$  bits, whereas in Section 5.3 we present deterministic extractors of similar performance. Both extractors work in  $\mathcal{AC}^0$ , which is optimal in light of the results presented in Section 5.2 (assuming that  $\ell \leq \text{poly}(\log n)$ ). Specifically, in Section 5.2, we prove that there exist no strong  $\mathcal{AC}^0$  extractors for min-entropy lower than  $n/\text{poly}(\log n)$  (regardless of the seed length). We also show that the same holds for ordinary extractors that output  $(1 + \Omega(1)) \cdot r(n)$  bits when using a seed of length  $r(n)$ .

## 5.1 Extraction with a logarithmically long seed

Although extraction from bit-fixing sources is possible without using a random seed (see [18, 41]), the known deterministic (i.e., seedless) extractors are not computable in  $\mathcal{AC}^0$ . In this section we present seeded extractors (for bit-fixing sources) that are computable in  $\mathcal{AC}^0$  (and use seeds of logarithmic length). In Section 5.3, following ideas of Gabizon *et al.* [26], we shall use the aforementioned seeded extractors to obtain deterministic extractors of similar performance, based on new deterministic extractors that extract poly-logarithmically many bits.

► **Theorem 5.2** (seeded extractor for block-fixing sources). *Let  $k : \mathbb{N} \rightarrow \mathbb{N}$  and  $\epsilon : \mathbb{N} \rightarrow [0, 1]$  be such that  $k(n) \geq n/\text{poly}(\log n)$  and  $\epsilon(n) \geq 1/\text{poly}(n)$ . Then, there exists a function  $E : \{0, 1\}^{n\ell} \times \{0, 1\}^{O(\log n)} \rightarrow \{0, 1\}^{\text{poly}(k(n)/n) \cdot n\ell}$  that is computable in uniform  $\mathcal{AC}^0$  and constitutes a strong  $\epsilon$ -error extractor for  $(n, k, \ell)$ -block-fixing sources.*

Theorem 5.2 is related to results of Gabizon *et al.* [26, Sec. 6], but the parameters are quite different: Most importantly, their extractors are not in  $\mathcal{AC}^0$ , although they could have obtained such extractors with a different setting of the parameters in their construction.

<sup>18</sup> Indeed, such sources are sometimes called *oblivious bit-fixing sources*, in order to distinguish them from non-oblivious bit-fixing sources [41] in which the remaining  $n - k$  bits are fixed as a function of the values of the  $k$  random bits.



Likewise, they extract only  $m(n) = k(n)^{\Omega(1)}$  bits. On the other hand, they treat any  $k(n) \geq \text{poly}(\log n)$  and use a seed of length  $O(\log k(n))$ . The pivot of our solution is the use of a new pseudorandom partition generator, captured by Lemma 5.3.

**Proof.** For sake of simplicity, we consider the case of  $\ell = 1$ , and denote the bits of the source by  $x_1 \circ \dots \circ x_n$ . The basic idea, which goes back to Gabizon *et al.* [26, Sec. 6], is to use a random partition of the source into many short sources, and extract bits from each of these short sources by XORing the corresponding bits. That is, we use the seed of the extractor to generate (pseudo)random subsets, denoted  $S_1, \dots, S_m \subset [n]$ , and output the bits  $\bigoplus_{j \in S_i} x_j$  for  $i = 1, \dots, m$ . This works provides that each of the  $S_i$ 's contains a location of a random (i.e., non-fixed) bit of the original source, and that these (non-fixed) locations are distinct. Furthermore, the  $S_i$ 's should each be of size  $\text{poly}(\log n)$  so to allow for the XOR to be implemented by constant-depth  $\text{poly}(n)$ -size circuits.

The first idea that comes to mind is to generate the subsets by repeatedly invoking an ordinary sampler (cf. [28]), using related seeds, but this seems to require that  $m < \sqrt{n}$ . An alternative approach, which also goes back to Gabizon *et al.* [26, Sec. 6], is to generate a pseudorandom partition of  $[n]$  into  $m$  subsets of equal size. Unfortunately, the techniques used in [26, Sec. 5] also seems to require that  $m < \sqrt{n}$ , whereas we seek  $m = n/\text{poly}(\log n)$ . Furthermore, we need to generate such a partition using a seed of logarithmic length and each subset in the partition should have a strong hitting property (i.e., as stated in the second condition of the following claim). (We comment that similar problems arise in the proof of Theorems 3.2 and 3.8, but the parameters and hitting requirements there are different.)

► **Lemma 5.3** (pseudorandom partitions with a strong hitting property). *For  $\delta, \gamma > 0$  and  $n, t \in \mathbb{N}$  such that  $t = \Theta(\delta^{-1} \log(1/\gamma))^2$  divides  $n$ , let  $m = n/t$  and  $r = O(\log(n/\gamma))$ . Then, there exists an explicit function  $G : \{0, 1\}^r \rightarrow ([n]^t)^m$  such that the following two conditions hold.*

1. *For every  $u \in \{0, 1\}^r$  the  $m$ -sequence  $G(u)$  is a partition of  $[n]$ ; that is, for every  $j_1 \neq j_2$  it holds that  $G(u)_{j_1}$  and  $G(u)_{j_2}$  are disjoint  $t$ -subsets of  $[n]$ .*
2. *For every  $T \subseteq [n]$  of density  $\delta$ , with probability at least  $1 - \gamma$ , each subset in  $G(U_r)$  hits  $T$ ; that is,  $\Pr[\exists j \in [m] \text{ s.t. } G(U_r)_j \cap T = \emptyset] \leq \gamma$ .*

**Proof.** Our starting point is an ordinary hitter  $H : \{0, 1\}^{r'} \rightarrow [n]^s$ , where  $r' = O(\log(n/\gamma'))$  and  $s = O(\delta^{-1} \log(1/\gamma'))$ , that hits each set of density  $\delta$  with probability at least  $1 - \gamma'$ . We further assume that every two sample points of this hitter are uniformly distributed independently of one another. Note that the combined hitter of [28, Apdx. C.3] satisfies the first requirement, whereas the second requirement can be achieved by “randomizing” the original sample via a sequence of pairwise independent “shifts” (cf. Claim 3.4).<sup>19</sup> Associating

<sup>19</sup> Alternatively, we can apply the sampler used in the proof of Theorem 3.2 (with  $c = 1$ ), but set the parameters in order to satisfy hitting rather than sampling. The difference between the two versions is that shifting the samples (as proposed in the main text) is different from shifting the seeds used to generate the subsamples. Specifically, recall that the combined hitter of [28, Apdx. C.3] has the form  $H''(u) = \cup_{i \in [t'']} H'_i(v_i)$ , where  $(v_1, \dots, v_{t''}) \leftarrow W(u)$  is a random walk on an expander and  $H'_i$  is a pairwise independence generator. In the main text we suggested using pairwise independent shifts of the sample  $\cup_{i \in [t'']} H'(v_i)$ , whereas in the proof of Theorem 3.2 we used the sample  $\cup_{i \in [t'']} H'(v'_i)$ , where  $(v'_1, \dots, v'_{t''})$  is obtained by a pairwise independent shift of  $(v_1, \dots, v_{t''})$ . In the analysis of the first alternative one relies on the generalized hitting property of  $H'$ , whereas in the analysis of the second alternative one relies on the generalized hitting property of  $W$ , where **generalized hitting** refers to generating a sequence  $(\sigma_1, \dots, \sigma_s)$  such that for every sequence of sets  $(T_1, \dots, T_s)$  (each of density  $\delta$ ) with probability at least  $1 - \gamma$  there exists an  $i \in [s]$  such that  $\sigma_i \in T_i$ . Note that the standard analyses of both  $H'$  and  $W$  extends to this case.



$[n]$  with  $[t] \times \mathbb{Z}_m$  and using  $t = s^2$ , it follows that with probability at least half, the sample  $((i_1, j_1), \dots, (i_s, j_s)) \leftarrow H(U_{r'})$  contains no collision on the first coordinate (i.e., for every  $a \neq b$  it holds that  $i_a \neq i_b$ ). Hence, conditioned on this event, for every set  $T$  of density  $\delta$ , the probability that  $H(U_{r'})$  hits  $T$  is at least  $1 - 2\gamma'$ . Consider the following generator  $G' : \{0, 1\}^{r'} \rightarrow ([n]^t)^m$ :

1. On input  $u \in \{0, 1\}^{r'}$ , obtain  $((i_1, j_1), \dots, (i_s, j_s)) \leftarrow H(u)$ . If there exist  $a \neq b$  such that  $i_a = i_b$ , then output a fixed partition  $(S_0, \dots, S_{m-1})$  of  $[n] \equiv [t] \times \mathbb{Z}_m$  into  $t$ -subsets; for example,  $S_j = \{(i, j) : i \in [t]\}$ . In this case we say that  $u$  is bad.
2. Otherwise (i.e., for every  $a \neq b$  it holds that  $i_a \neq i_b$ ), for every  $i \in [t]$  and  $j \in \mathbb{Z}_m$ , let  $p_{i,j} = (i, j_a + j \bmod m)$  if  $i = i_a$  and  $p_{i,j} = (i, j)$  otherwise (i.e.,  $i \notin \{i_1, \dots, i_s\}$ ). For every  $j \in \mathbb{Z}_m$ , define  $G'_j(u) = \{p_{i,j} : i \in [t]\}$ , and output the  $m$ -sequence  $(G'_0(u), \dots, G'_{m-1}(u))$ . In this case we say that  $u$  is good, and it holds that

$$G'_0(u) = \{(i_a, j_a) : a \in [s]\} \cup \{(i, 0) : i \in ([t] \setminus \{i_1, \dots, i_s\})\}$$

$$\text{and } G'_j(u) = \{(i, j' + j \bmod m) : (i, j') \in G'_0(u)\}.$$

Note that in each case the output sequence is a partition of  $[n] \equiv [t] \times \mathbb{Z}_m$ . In the “good case” (i.e., of a good  $u$ ) this follows since  $G'_0(u) = \{(i, g(i)) : i \in [t]\}$  for some function  $g : \mathbb{Z}_m \rightarrow \mathbb{Z}_m$  (i.e.,  $g(i_a) = j_a$  for  $a \in [s]$  and  $g(i) = 0$  for  $i \notin \{i_1, \dots, i_s\}$ ) and  $G'_j(u) = \{(i, g(i) + j \bmod m) : i \in [t]\}$ . The foregoing properties of  $H$  imply that, with probability at least half  $U_{r'}$  is good, whereas conditioned on this event the probability that  $G'_j(U_{r'}) \in T$  is at least  $1 - 2\gamma'$ , for every set  $T$  of density  $\delta$  and for every  $j \in \mathbb{Z}_m$ .

Let us reflect for a moment on the structure of the generator  $G'$ . It uses a fixed partition of  $[n] \equiv [t] \times \mathbb{Z}_m$  into  $t$  cycles, each of length  $m$ , where the  $i^{\text{th}}$  cycle consists of  $\{(i, j) : j \in \mathbb{Z}_m\}$ . Assuming that the sample  $H(u) \in ([t] \times \mathbb{Z}_m)^s$  hits exactly  $s$  cycles, we augmented the sample to a cover of all cycles by adding (dummy) elements of the form  $(i, 0)$  for every uncovered cycle. (Of course, this cannot damage the hitting property.) Finally, we use the  $m$  “shifts” of the resulting set  $S_0$  as a partition, where the  $j^{\text{th}}$  shift of the set  $S_0 \subset [t] \times \mathbb{Z}_m$  equals  $\{(i, j' + j) : (i, j') \in S_0\}$ . In case the initial sample hits less than  $s$  cycles (i.e., some cycle is hit by more than a single point in the sample), the generator outputs a fixed partition. The only problem is that the latter bad event may occur with constant probability (of at most  $1/2$ ), whereas we want it to occur (in the final generator) with probability smaller than  $\gamma$ .

The desired generator  $G$  is obtained by taking a random walk of length  $t' = O(\log(1/\gamma))$  on a constant degree expander with vertex set  $\{0, 1\}^{r'}$ , and outputting  $G'(v)$  where  $v$  is the first vertex on the walk that is good. If no such vertex exists, then we just output the fixed partition, but this event occurs with probability at most  $\gamma/2$ .

Denoting the random walk by  $(v_1, \dots, v_{t'})$ , recall that the probability that some  $v_j$  is good is at least  $1 - \exp(-\Omega(t')) > 1 - (\gamma/2)$ . We show that conditioned on this event, for every  $T$  of density  $\delta$ , with probability at least  $1 - (3\gamma/4) - (8t'm\gamma'/\gamma)$ , each set in the partition output by  $G$  hits the set  $T$ . This is shown by considering only the indices  $j \in [t']$  such that with probability at least  $\gamma/4t'$  vertex  $v_j$  is the first vertex in the walk that is good. Specifically, denote by  $\mathcal{G}_j$  the event that  $v_j$  is good, and let  $\mathcal{G}'_j$  denote the event  $\mathcal{G}_j \wedge \neg(\bigvee_{j' < j} \mathcal{G}_{j'})$ . Let  $J = \{j \in [t'] : \Pr[\mathcal{G}'_j] \geq \gamma/4t'\}$  and note that  $\Pr[\bigvee_{j \in J} \mathcal{G}'_j] > 1 - (\gamma/2) - t' \cdot (\gamma/4t') = 1 - 3\gamma/4$ . On the other hand, for each  $j \in J$ , the probability that each set in  $G(v_j)$  hits  $T$  is at least  $1 - 2m\gamma'$ . Let us denote this event by  $\mathcal{H}_j$ . Then, the probability that all sets output by  $G$

hit  $T$  is at least

$$\begin{aligned} \sum_{j \in J} \Pr[\mathcal{G}'_j] \cdot \Pr[\mathcal{H}_j | \mathcal{G}'_j] &\geq \Pr[\bigvee_{j \in J} \mathcal{G}'_j] \cdot \min_{j \in J} \{\Pr[\mathcal{H}_j | \mathcal{G}'_j]\} \\ &\geq \left(1 - \frac{3\gamma}{4}\right) \cdot \min_{j \in J} \left\{1 - \frac{\Pr[\neg \mathcal{H}_j]}{\Pr[\mathcal{G}'_j]}\right\}. \end{aligned}$$

Using  $\Pr[\neg \mathcal{H}_j] \leq 2m\gamma'$  and  $\Pr[\mathcal{G}'_j] \geq \gamma/4t'$  (for any  $j \in J$ ), we obtained the claimed lower bound of  $(1 - 3\gamma/4) \cdot (1 - (2m\gamma')/(\gamma/4t')) > (1 - 3\gamma/4) + (8t'm\gamma'/\gamma)$ .

Setting  $\gamma' = \gamma^2/32t'm$ , it follows that each set output by  $G$  hits a set  $T$  of density  $\delta$  with probability at least  $1 - \gamma$ . Noting that  $G$  uses a seed of length  $r = r' + O(t') = O(\log(n/\gamma))$ , the lemma follows. ◀

**Wrapping up.** We set  $\delta = k(n)/n = 1/\text{poly}(\log n)$  and  $\gamma = \epsilon = 1/\text{poly}(n)$ , and use Lemma 5.3 with  $s = O(\delta^{-1} \log(1/\gamma))$  and  $t = s^2 = O(\delta^{-1} \log n)^2$ , which implies  $m = n/t = \Omega(\delta^2 n / \log^2 n) = n/\text{poly}(\log n)$  and  $r = O(\log n)$ . The extractor  $E(x, u)$  outputs  $y_1 \circ \dots \circ y_m$  such that  $y_j = \bigoplus_{i \in G(u)_{j+1}} x_i$ . Since  $G$  can be computed by uniform depth-two circuits of size  $\text{poly}(n)$  and  $t = \text{poly}(\log n)$ , it follows that  $E$  is in uniform  $\mathcal{AC}^0$ . In analyzing the performance of  $E$  on an arbitrary  $(n, k)$ -bit-fixing source, let  $T$  denote the set of  $k$  unfixed bits in the source, and note that the output bits (i.e.,  $y_j$ 's) depend on disjoint sets of bits in the source and that with probability at least  $1 - \gamma$  each output bit depends on some unfixed bit of the source. The theorem follows for  $\ell = 1$ , and the argument for general  $\ell$  is identical. ◀

## 5.2 Impossibility results

The following impossibility result asserts the optimality of Theorem 5.2 with respect to the parameter  $k$  (i.e., the number of random blocks in the source): While Theorem 5.2 asserts strong extractors that are computable in  $\mathcal{AC}^0$  for any  $k(n) = n/(\log n)^{O(1)}$ , the following result asserts that this is not possible for any  $k(n) = n/(\log n)^{\omega(1)}$ .

► **Theorem 5.4** (impossibility of strong extraction in  $\mathcal{AC}^0$ ). *Suppose that  $E : \{0, 1\}^{n\ell} \times \{0, 1\}^r \rightarrow \{0, 1\}$  is computable by  $s(n)$ -size circuits of depth  $d = d(n)$ . If  $E$  is a strong 0.499-error extractor for  $(n, k, \ell)$ -block-fixing sources, then  $k > n/(\ell \cdot O(\log s(n))^{d-1})$ .*

Recall that  $\ell = 1$  corresponds to bit-fixing sources. Note that the current result regarding  $(n, k)$ -bit-fixing sources implies that *if a strong  $(0.499/\ell)$ -error extractor for  $(n, k, \ell)$ -block-fixing sources is computable in  $\mathcal{AC}^0$ , then  $k \geq n/\text{poly}(\log n)$ . We conjecture that this holds even for 0.499-error extractor for  $(n, k, \ell)$ -block-fixing sources; that is, we conjecture that the linear dependence on  $\ell$ , in the foregoing results, can be eliminated.*

Before proving Theorem 5.4 we note that it is incomparable but related to Theorem 1.1 (i.e., Viola's [57, Thm. 6.4]): Theorem 5.4 refers to strong extractors that output a single bit, whereas Theorem 1.1 applies to ordinary extractors that output a constant factor more bits than their seed length. (A version that refers to ordinary extractors is presented later; see Theorem 5.5.) An important advantage of Theorem 5.4 over Theorem 1.1 is that it refers to a much more restricted class of sources (i.e.,  $(n, k, 1)$ -block-fixing sources rather than  $(n, k)$ -sources).

**Proof.** For simplicity, we start with the case of  $\ell = 1$ . Fixing any value  $\sigma \in \{0, 1\}^r$  of the seed, consider the residual depth  $d$  circuit of size  $s = s(n)$ , denoted  $C_\sigma$ , that computes the

mapping  $x \mapsto E(x, \sigma)$ . Invoking the “average sensitivity bound” of Linial *et al.* [39], as improved by Boppana [12], for every  $C_\sigma$  we have

$$\sum_{i \in [n]} \mathbb{I}_i(C_\sigma) < B \stackrel{\text{def}}{=} O(\log s)^{d-1}, \quad (5)$$

where for any Boolean function  $f : \{0, 1\}^n \rightarrow \{0, 1\}$

$$\mathbb{I}_i(f) \stackrel{\text{def}}{=} \Pr_{x \leftarrow U_n} [f(x) \neq f(x \oplus 0^{i-1} 10^{n-i})]. \quad (6)$$

(See background in [37, Sec. 12.4] and [47, Chap. 2].)

It follows that there exists a set of  $\epsilon n/B$  indices  $I \subseteq [n]$  such that  $\mathbf{E}_{\sigma \leftarrow U_r} [\sum_{i \in I} \mathbb{I}_i(C_\sigma)] < \epsilon$ , since the expectation over all  $|I|$ -sized subsets is smaller than  $\epsilon$ . Furthermore, there exists a string  $z \in \{0, 1\}^n$  such that

$$\mathbf{E}_{\sigma \leftarrow U_r} \left[ \sum_{i \in I} \Pr_{x \leftarrow U_n} [C_\sigma(x) \neq C_\sigma(x \oplus 0^{i-1} 10^{n-i}) \mid x_{[n] \setminus I} = z_{[n] \setminus I}] \right] < \epsilon \quad (7)$$

and there exists a set  $G \subseteq \{0, 1\}^r$  of density at least  $1 - \sqrt{\epsilon}$  such that for every  $\sigma \in G$  it holds that

$$\sum_{i \in I} \Pr_{x \leftarrow U_n} [C_\sigma(x) \neq C_\sigma(x \oplus 0^{i-1} 10^{n-i}) \mid x_{[n] \setminus I} = z_{[n] \setminus I}] < \sqrt{\epsilon}. \quad (8)$$

Fixing  $I$  and  $z$  as above, consider an  $(n, |I|, 1)$ -block-fixing source  $X = (X_1, \dots, X_n)$  such that  $X_i = z_i$  if  $i \in [n] \setminus I$  and  $X_i$  is random otherwise. We next show that, for every  $\sigma \in G$  there exists a bit  $y_\sigma$  so that  $\Pr[C_\sigma(X) = y_\sigma] > 1 - \sqrt{\epsilon}$ ,

To prove the above claim, assume that  $p \stackrel{\text{def}}{=} \Pr[C_\sigma(X) = y_\sigma] \leq 1 - \sqrt{\epsilon}$  and  $p \geq 1/2$  (w.l.o.g.). Then,  $\Pr_{x, y \leftarrow U_n} [C_\sigma(x) \neq C_\sigma(y) \mid x_{[n] \setminus I} = y_{[n] \setminus I} = z_{[n] \setminus I}] = 2p(1-p) \geq \sqrt{\epsilon}$ . This implies that there exists  $s \in \{0, 1\}^n$  such that  $s_{[n] \setminus I} = 0^{n-|I|}$  and  $\Pr[C_\sigma(X) \neq C_\sigma(X \oplus s)] \geq \sqrt{\epsilon}$ , which contradicts Eq. (8), since

$$\begin{aligned} \Pr[C_\sigma(X) \neq C_\sigma(X \oplus s)] &\leq \sum_{i: s_i=1} \Pr[C_\sigma(X) \neq C_\sigma(X \oplus 0^{i-1} 10^{n-i})] \\ &\leq \sum_{i \in I} \Pr[C_\sigma(X) \neq C_\sigma(X \oplus 0^{i-1} 10^{n-i})]. \end{aligned}$$

(The first inequality uses the fact that  $X \equiv X \oplus s'$  for every  $s' \in \{0, 1\}^n$  such that  $s'_{[n] \setminus I} = 0^{n-|I|}$ .)

We have established that for every  $\sigma \in G$ , it holds that  $\Pr[E(X, \sigma) = y_\sigma] > 1 - \sqrt{\epsilon}$ , for some bit  $y_\sigma$ , whereas  $\Pr[U_1 = y_\sigma] = 1/2$ . It follows that  $\Delta[E(X, U_r) \circ U_r; U_1 \circ U_r]$ , which equals  $\mathbf{E}_{u \leftarrow U_r} [\Delta[E(X, u); U_1]]$ , is greater than  $\Pr[U_r \in G] \cdot (1 - \sqrt{\epsilon} - 0.5) \geq (1 - \sqrt{\epsilon}) \cdot (0.5 - \sqrt{\epsilon}) > 0.5 - 2\sqrt{\epsilon}$ . Hence, if  $E$  is a strong  $(0.5 - 2\sqrt{\epsilon})$ -error extractor for  $(n, k, 1)$ -block-fixing sources, then  $k > \epsilon n/B$ .

The argument for general  $\ell > 1$  proceeds analogously, except that here we have  $n \cdot \ell$  variables/indices, which are partitioned into  $n$  blocks. We first consider the set  $L$  of all indices in  $[n] \times [\ell]$  such that for every  $(i, j) \in L$  it holds that  $\mathbf{E}_{\sigma \leftarrow U_r} [\mathbb{I}_{i,j}(C_\sigma)] < 2B/n$ , where  $B = O(\log s)^{d-1}$  (as in Eq. (5)). Recalling that  $\sum_{(i,j) \in [n] \times [\ell]} \mathbf{E}_{\sigma \leftarrow U_r} [\mathbb{I}_{i,j}(C_\sigma)] < B$ , it follows that  $|L| \geq n\ell - n/2$ . We consider the set  $L' \subseteq [n]$  of blocks such that  $i \in L'$  if for every  $j \in [\ell]$  it holds that  $(i, j) \in L$ . Then,  $|L'| \geq n/2$ . We now select an arbitrary set  $I' \subseteq L'$  of size  $\epsilon n/2\ell B$ , let  $I = I' \times [\ell]$ , and proceed as before, while noting that (as

before) it holds that  $\mathbf{E}_{\sigma \leftarrow U_r} [\sum_{(i,j) \in I} \mathbb{I}_{i,j}(C_\sigma)] < \epsilon$ , since  $|I| \cdot 2B/n = \epsilon$ . That is, we fix  $I$ ,  $z = (z_1, \dots, z_n) \in (\{0, 1\}^\ell)^n$  and  $G$  (as before), and note that for every  $\sigma \in G$  it holds that

$$\sum_{(i,j) \in I} \Pr_{x \leftarrow U_{n\ell}} \left[ C_\sigma(x) \neq C_\sigma(x \oplus 0^{\text{id}\mathbf{x}(i,j)-1} 10^{n-\text{id}\mathbf{x}(i,j)}) \mid x_{[n] \setminus I} = z_{[n] \setminus I} \right] < \sqrt{\epsilon}, \quad (9)$$

where  $\text{id}\mathbf{x}(i, j) = (i - 1) \cdot \ell + j$ . Now, consider an  $(n, |I|, \ell)$ -block-fixing source  $X = (X_1, \dots, X_n)$  such that  $X_i = z_i$  if  $i \in [n] \setminus I'$  and  $X_i$  is random (i.e., distributed as  $U_\ell$ ) otherwise. Then, for every  $\sigma \in G$  there exists a bit  $y_\sigma$  so that  $\Pr[C_\sigma(X) = y_\sigma] > 1 - \sqrt{\epsilon}$ , since otherwise Eq. (9) is contradicted. The claim follows as before, but note that  $|I'| = |I|/\ell = \epsilon n/2\ell B$ .  $\blacktriangleleft$

**► Theorem 5.5** (impossibility of ordinary extraction in  $\mathcal{AC}^0$ ). *Suppose that  $E : \{0, 1\}^n \times \{0, 1\}^r \rightarrow \{0, 1\}^m$  is computable by  $s(n)$ -size circuits of depth  $d = d(n)$ , and let  $m' = m - r$ . Suppose that  $\delta > 0$  satisfies  $\binom{m}{\lfloor \delta m' \rfloor} < 1 + \delta \cdot 2^{m'}$ , which is satisfied whenever either  $\delta < 1/m'$  or  $\delta \leq 1/3 \log m$ . Then, if  $E$  is a  $(1 - 2\delta - 2^{-m'})$ -error extractor for  $(n, k, 1)$ -block-fixing sources, then  $k > \frac{\delta^3 m' n}{m \cdot O(\log s(n))^{d-1}}$ . In particular:*

1. *If  $m = r + 1$  (i.e.,  $m' = 1$ ) and  $E$  is a 0.499-error extractor for  $(n, k, 1)$ -block-fixing sources, then  $k > \frac{n}{m \cdot O(\log s(n))^{d-1}}$ .*
2. *If  $m = r + \Omega(r)$  (i.e.,  $m' = \Omega(r)$ ) and  $E$  is a 0.999-error extractor for  $(n, k, 1)$ -block-fixing sources, then  $k > n/O(\log s(n))^{d-1}$ .*

Setting  $\delta = \min(o(1), 1/3 \log m)$ , the general case implies that *if  $E$  is a  $(1 - 2^{-m'} - o(1))$ -error extractor for  $(n, k, 1)$ -block-fixing sources, then  $k > \frac{m' n}{O(m) \cdot O(\log s(n))^{d-1}}$ . Theorem 5.5 generalizes Theorem 1.1, which only refers to the case of  $m' = \Omega(r)$ . Another important advantage of Theorem 5.5 over Theorem 1.1 is that it refers to a much more restricted class of sources (i.e.,  $(n, k, 1)$ -block-fixing sources rather than  $(n, k)$ -sources).<sup>20</sup>*

**Proof.** The proof is very similar to the proof of Theorem 5.4, except that we consider  $2^r \cdot m$  residual circuits  $C_{\sigma,j}$  such that  $C_{\sigma,j}(x)$  computes the  $j^{\text{th}}$  bit of  $E(x, \sigma)$ . Specifically, we again derive a set of  $\epsilon n/B$  indices  $I \subseteq [n]$  and a string  $z \in \{0, 1\}^n$  such that

$$\mathbf{E}_{\sigma \leftarrow U_r, j \in_R [m]} \left[ \sum_{i \in I} \Pr_{x \leftarrow U_n} \left[ C_{\sigma,j}(x) \neq C_{\sigma,j}(x \oplus 0^{i-1} 10^{n-i}) \mid x_{[n] \setminus I} = z_{[n] \setminus I} \right] \right] < \epsilon, \quad (10)$$

where  $j \in_R [m]$  denotes that  $j$  is distributed uniformly in  $[m]$ . We shall again fix  $I$  and  $z$  as above, and consider an  $(n, |I|, 1)$ -block-fixing source  $X = (X_1, \dots, X_n)$  such that  $X_i = z_i$  if  $i \in [n] \setminus I$  and  $X_i$  is random otherwise. We set  $\epsilon = \delta^3 m'/m$ , and denote  $C_\sigma(x) = E(x, \sigma)$ . Now, letting  $\text{dist}(y, z)$  denote the Hamming distance between the  $m$ -bit long strings  $y$  and  $z$  (i.e.,  $\text{dist}(y_1 \dots y_m, z_1 \dots z_m) = |\{i \in [m] : y_i \neq z_i\}|$ ), we get

$$\mathbf{E}_{\sigma \leftarrow U_r} \left[ \sum_{i \in I} \mathbf{E}_{x \leftarrow X} [\text{dist}(C_\sigma(x), C_\sigma(x \oplus 0^{i-1} 10^{n-i}))] \right] < m \cdot \epsilon = \delta^3 m'. \quad (11)$$

<sup>20</sup> We mention that the preceding version of [57] contains a result that is more closely related to Theorem 5.5: The  $(n, k)$ -source used in the proof there also belongs to a very restricted class; specifically, the source is a randomized process that produces an output by starting with  $0^n$  and taking  $k/36$  steps such that at each step a random position is selected (with repetitions) and its value is flipped.

Hence, there exists a set  $G \subseteq \{0, 1\}^r$  of density at least  $1 - \delta$  such that for every  $\sigma \in G$  it holds that

$$\sum_{i \in I} \mathbf{E}_{x \leftarrow X} [\text{dist}(C_\sigma(x), C_\sigma(x \oplus 0^{i-1}10^{n-i}))] < \delta^2 m'. \quad (12)$$

Then, for every  $\sigma \in G$  there exists a string  $y_\sigma \in \{0, 1\}^m$  so that  $\mathbf{E}[\text{dist}(C_\sigma(X), y_\sigma)] < \delta^2 m'$ , because for two independent samples  $x$  and  $x'$  drawn from  $X$ , it holds that

$$\begin{aligned} & \mathbf{E}_{x, x'} [\text{dist}(C_\sigma(x), C_\sigma(x'))] \\ & \leq \max_{s \in \{0, 1\}^n : s_{[n] \setminus I} = 0^{n-|I|}} \left\{ \mathbf{E}_{x \leftarrow X} [\text{dist}(C_\sigma(x), C_\sigma(x \oplus s))] \right\} \\ & \leq \max_{s : s_{[n] \setminus I} = 0^{n-|I|}} \left\{ \sum_{i : s_i = 1} \mathbf{E}_{x \leftarrow X} [\text{dist}(C_\sigma(x), C_\sigma(x \oplus 0^{i-1}10^{n-i}))] \right\} \end{aligned}$$

which (by Eq. (12)) is smaller than  $\delta^2 m'$ . Hence, for every  $\sigma \in G$ , it holds that  $\Pr[\text{dist}(C_\sigma(X), y_\sigma) \leq \delta m'] \geq 1 - \delta$ . Defining  $S = \{y_\sigma : \sigma \in G\}$ , we note that, with probability at least  $(1 - \delta)^2$ , the Hamming distance between  $E(X, U_r)$  and  $S$  (i.e., the distance to the closest string in  $S$ ) is at most  $\lfloor \delta m' \rfloor$ . On the other hand, the probability that  $U_m$  is at Hamming distance at most  $\lfloor \delta m' \rfloor$  from  $S$  is upper bounded by  $\frac{\binom{m}{\lfloor \delta m' \rfloor} \cdot |S|}{2^m} < 2^{-m'} + 4\delta$ , since  $|S| \leq 2^r = 2^{m-m'}$  and  $\binom{m}{\lfloor \delta m' \rfloor} < 1 + \delta \cdot 2^{m'}$ . It follows that  $\Delta[E(X, U_r); U_m] > (1 - \delta) - (2^{-m'} + \delta)$ , whereas  $X$  has  $\delta^3 m' n / mB$  random bits.  $\blacktriangleleft$

### 5.3 Deterministic extractors

Recall that the bit-fixing model allows for deterministic extractors (e.g.,  $E(x) = \bigoplus_{i \in [n]} x_i$ ), which work for even lower min-entropy rate than those that are impossible for  $\mathcal{AC}^0$ , but indeed these extractors are not computable in  $\mathcal{AC}^0$ . Still, it is possible that whenever extraction in  $\mathcal{AC}^0$  is possible for bit-fixing sources, this is also possible via deterministic extractors. We show that this is indeed the case. Our proof proceeds in three steps: First, we present  $\mathcal{AC}^0$  circuits that extract a single bit (see Theorem 5.8), next we use them to present  $\mathcal{AC}^0$  circuits that extract poly-logarithmically many bits (see Theorem 5.15), and finally we combine these with the extractor of Theorem 5.2 to extract  $n/\text{poly}(\log n)$  bits.

Recall that a **deterministic extractor** (a.k.a seedless extractor) is one that gets no seed (i.e., has seed length  $r(n) \equiv 0$ ). By definition, any deterministic extractor is strong. An obvious deterministic extractor for  $(n, k)$ -bit-fixing sources, which is implementable in  $\mathcal{AC}^0$  when  $k(n) \geq n - \text{poly}(\log n)$ , is  $E(x) = \bigoplus_{i \in [n-k(n)+1]} x_i$ . A first indication that one can do much better is provided by Ajtai and Linial's non-explicit construction of  $\mathcal{AC}^0$  circuits in which "large sets have small influence" [4, Sec. 5]. As shown in [41, Lem. 6.1], such circuits are deterministic extractors for *non-oblivious bit-fixing sources*.

**► Definition 5.6** (non-oblivious bit-fixing sources [41]). A non-oblivious  $(n, k)$ -bit-fixing source is a sequence of random variables  $X = (X_1, \dots, X_n) \in \{0, 1\}^n$  such that there exists a set of at least  $k$  indices  $I \subseteq [n]$  and a sequence of functions  $f_1, \dots, f_n : \{0, 1\}^{|I|} \rightarrow \{0, 1\}$  such that  $X_I$  is uniformly distributed over  $\{0, 1\}^{|I|}$  and  $X_i = f_i(X_I)$  for every  $i \in [n] \setminus I$ .

Bit-fixing sources as in Definition 5.1 are a special case in which the  $f_i$ 's are constant functions. Such sources are sometimes called *oblivious bit-fixing sources*. Clearly, any  $\epsilon$ -extractor for non-oblivious  $(n, k)$ -bit-fixing sources is an  $\epsilon$ -extractor for (oblivious)  $(n, k)$ -bit-fixing sources.

► **Theorem 5.7** (deterministic extraction in  $\mathcal{AC}^0$  for non-oblivious bit-fixing sources [4, Sec. 5]). *There exist  $\mathcal{AC}^0$  circuits  $C : \{0, 1\}^n \rightarrow \{0, 1\}$  such that, for every  $k$ , it holds that  $C$  is a deterministic  $O((1 - k/n) \cdot \log^2 n)$ -error extractors for non-oblivious  $(n, k)$ -bit-fixing sources.*

As shown in [41, Lem. 6.1&6.2], the error probability of extractors of the foregoing type is captured by the notion of *influence of sets*. For a Boolean function  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  and a set  $S \subset [n]$ , the influence of  $S$  on  $f$ , denoted  $I_S(f)$ , is defined as the maximum of  $\Pr_{x \leftarrow U_n}[f(x) \neq f(g(x))]$ , taken over all functions  $g : \{0, 1\}^n \rightarrow \{0, 1\}^n$  that satisfy  $g(x)_{[n] \setminus S} = x_{[n] \setminus S}$  for every  $x \in \{0, 1\}^n$  (i.e.,  $g$  only changes the values of  $x$  at location in  $S$  and does so depending on the entire input).<sup>21</sup> Ajtai and Linal [4, Sec. 5] proved that there exist balanced  $\mathcal{AC}^0$  circuits  $C : \{0, 1\}^n \rightarrow \{0, 1\}$  such that the influence of every set of density  $\rho$  on  $C$  is  $O(\log^2 n) \cdot \rho$ , where  $f$  is balanced if  $\Pr[f(U_n) = 1] = 0.5$ . Hence, these circuits are deterministic  $O(\rho \log^2 n)$ -error extractors for the corresponding set of sources (i.e., non-oblivious  $(n, n - \rho n)$ -bit-fixing sources).

Theorem 5.7 is meaningful only for min-entropy rate approaching 1; that is, it is only meaningful for non-oblivious  $(n, k)$ -bit-fixing sources with  $k \geq n - O(n/\log^2 n)$ . This is not an artifact of the proof (nor even of the fact that the extractor is in  $\mathcal{AC}^0$ ): As shown by Kahn, Kalai, and Linal [40] any deterministic  $\epsilon$ -extractor for *non-oblivious*  $(n, k)$ -bit-fixing sources must satisfy  $k \geq n - \Omega(\epsilon^{-1}n/\log n)$ . However, for *oblivious*  $(n, k)$ -bit-fixing sources, one can achieve a min-entropy rate that approaches 0; that is, a deterministic  $\mathcal{AC}^0$ -extractor for oblivious  $(n, k)$ -bit-fixing sources with any  $k \geq n/\text{poly}(\log n)$ .

► **Theorem 5.8** (deterministic extraction in  $\mathcal{AC}^0$  for bit-fixing sources). *For every  $k(n) \geq n/\text{poly}(\log n)$  and every  $\epsilon(n) > 1/\text{poly}(\log n)$ , there exist deterministic  $\epsilon$ -error extractors  $E : \{0, 1\}^n \rightarrow \{0, 1\}$  for  $(n, k)$ -bit-fixing sources such that the extractors are computable in  $\mathcal{AC}^0$ .*

Our proof of Theorem 5.8 builds upon Theorem 5.7, and thus inherits the non-uniformity of the latter. In addition, our own reduction of Theorem 5.8 to Theorem 5.7 is non-explicit, due to our use of a probabilistic analysis of the rank of rectangular matrices [10].

**Proof.** Our starting point is the extractor of non-oblivious bit-fixing sources asserted in Theorem 5.7. Denoting the corresponding  $\mathcal{AC}^0$  circuit by  $C : \{0, 1\}^n \rightarrow \{0, 1\}$ , our plan is to construct a new circuit  $C' : \{0, 1\}^{n'} \rightarrow \{0, 1\}$ , where  $n' = \tilde{O}(n)$ , by feeding each input of  $C$  with the parity of a random subset of  $\text{poly}(\log n)$  inputs of  $C'$ . Specifically,  $C'(x) = C(L_1(x), \dots, L_n(x))$ , where each  $L_i$  is a random linear function generated by selecting each  $x_i$  with probability  $p = \text{poly}(\log n)/n$ . Indeed, these  $L_i$ 's can be computed by a constant-depth circuits of size  $\text{poly}(n)$ .

Suppose that, for any choice  $V$  of  $\delta n' = k(n') \geq n$  variables (i.e.,  $x_i$ 's), at least  $n - \rho \cdot n$  of the linear functions (i.e.,  $L_i$ 's) are linearly independent as functions in the variables in  $V$ . Then, any fixing of  $n' - k(n')$  of the  $x$ -variables, leaves at least  $n - \rho n$  of the functions (i.e., the  $L_i$ 's) linearly independent, which means that assigning random values to  $k(n')$  of the inputs of  $C'$  (and setting the rest arbitrarily but obliviously of the random values) yields a random assignment to  $n - \rho n$  of the inputs of  $C$ .

In anticipation of considering the influence of sets of  $\rho n$  inputs on  $C$ , we set  $\rho = \epsilon/\Theta(\log n)^2 = 1/\text{poly}(\log n)$ , which guarantees that the influence of such sets is at most  $\epsilon$ . Next, we set  $m = \rho \cdot n$  and  $n' = \tilde{O}(n)$ , and seek a sparse  $n$ -by- $n'$  Boolean matrix  $M$  such

<sup>21</sup>The influence of a single variable, as defined in Eq. (6), is a special case: Indeed, when considering the influence of the variable  $i$ , it suffices to consider  $g(x) = x \oplus 0^{i-1}10^{n-i}$ .

that (1) each row of  $M$  has at most  $\text{poly}(\log n)$  one-entries, and (2) any  $n$ -by- $n$  sub-matrix of  $M$  has rank at least  $n - m$ . If we had such a matrix, then we can let its rows serve as the  $L_i$ 's and be done (for  $k(n') = n = n'/\text{poly}(\log n')$ ).

While we believe that such a matrix exists, we were not able to prove this conjecture. Instead, we take a somewhat longer route. First, we construct a matrix with properties as above over a finite field of quasi-polynomial (in  $n$ ) size. Then, we use this construction to get a matrix with such properties for a finite field of poly-logarithmic (in  $n$ ) size. Lastly, we show how to use the latter in our context. These three steps are captured by the following three claims.

► **Claim 5.9** (a desired matrix over a finite field of quasi-polynomial size). *For any  $m \in [n/\text{poly}(\log n), n]$  and  $n' \in [\Omega(n \log^2 n), \tilde{O}(n)]$ , and every finite field  $F$  of cardinality  $q \geq \exp(n'/m)$ , there exist an  $n$ -by- $n'$  matrix  $M$  over  $F$  that satisfies the following two properties.*

1. *Each row of  $M$  has at most  $\text{poly}(\log n)$  non-zero entries.*
2. *Each  $n$ -by- $n$  sub-matrix of  $M$  has rank at least  $n - m$ .*

**Proof.** Setting  $p = (\log n)/n$ , consider selecting a random sparse  $n$ -by- $n'$  matrix  $M$  over  $F$  by setting each entry to 0 with probability  $1 - p$ , and letting it be a uniformly distributed nonzero value otherwise. (The choices for the various entries are independent of one another.) Then, with probability at least  $1 - n \cdot \exp(-\Omega(pn')) \geq 1 - \exp(-\Omega(\log n)^3)$ , each row of  $M$  has  $\Theta(pn') = \text{poly}(\log n)$  non-zero entries.

In proving the second property, we use a result of Blomer *et al.* [10] that asserts that an  $n$ -by- $n$  matrix distributed as above has rank smaller than  $n - m$  with probability  $O(q^{-m})$ . Applying a union bound, we infer that the second property fails with probability at most

$$\begin{aligned} \binom{n'}{n} \cdot O(q^{-m}) &< (n^2)^n \cdot \exp(-\Omega(n \log^2 n)/m)^m \\ &= \exp(O(n \log n)) \cdot \exp(-\Omega(n \log^2 n)), \end{aligned}$$

where the inequality uses  $\log q \geq \Omega(n'/m) = \Omega(n \log^2 n)/m$ . The claim follows. ◀

► **Claim 5.10** (a desired matrix over a finite field of poly-logarithmic size). *For any  $m \in [n/\text{poly}(\log n), o(n)]$  and  $n' \in [\Omega(n \log^2 n), \tilde{O}(n)]$ , every finite field  $F'$  of cardinality  $q' = \text{poly}(n'/m)$ , and  $n'' = \text{poly}(q) \cdot n'$ , there exist an  $n$ -by- $n''$  matrix  $M'$  over  $F'$  that satisfies the following two properties.*

1. *Each row of  $M'$  has at most  $\text{poly}(\log n)$  non-zero entries.*
2. *Each  $n$ -by- $(2n/n') \cdot n''$  sub-matrix of  $M'$  has rank at least  $n - m$ .*

**Proof.** Let  $F$  be a finite field of size  $\exp(\Theta(n'/m))$  and  $M$  be a matrix as guaranteed by Claim 5.9. Let  $\ell = \lceil \log |F| \rceil = \Theta(n'/m) = \text{poly}(\log n)$ . For some  $q' = \text{poly}(n'/m)$  and  $\ell' = \text{poly}(q')$ , consider a linear error correcting code mapping  $\text{GF}(q')^\ell$  to  $\text{GF}(q')^{\ell'}$  such that this code has relative distance at least  $1 - (n/n')$ . (For example, the Reed-Solomon code of degree  $\ell$  over  $\text{GF}(q')$  uses  $\ell' = q'$  and has relative distance  $1 - (\ell/q')$ , whereas  $\ell/q' < n/n'$  provided that  $q' \geq (n'/m)^2$ .) Now, encode each element of  $F$  (viewed as an  $\ell$ -long sequence over  $\text{GF}(q')$ ) by the corresponding codeword, obtaining an  $n$ -by- $n' \ell'$  matrix  $M'$  over  $F' = \text{GF}(q')$ . We may assume that  $F'$  is a sub-field of  $F$ ; in fact, we should pick  $F$  to satisfy this condition (as well as the other conditions stated above).

Letting  $n'' = n' \cdot \ell' = \text{poly}(\log n) \cdot n$ , note that the number of nonzero entries in each row of  $M'$  is at most  $\text{poly}(\log n) \cdot \ell' = \text{poly}(\log n)$ , since each row of  $M$  has at most  $\text{poly}(\log n)$  non-zeros. This establishes the first property of  $M'$ .



To establish the second property of  $M'$ , consider an arbitrary choice of  $(2n/n') \cdot (n'\ell') = 2n\ell'$  columns of  $M'$ , and let  $M''$  denote the corresponding sub-matrix. Considering the partition of the columns of  $M'$  into  $n'$  blocks of length  $\ell'$  (each encoding a symbol of  $F$ ), we infer that at least  $n$  of these blocks contain more than  $\ell'n/n'$  chosen columns. Let us call these blocks heavy. Recalling that the linear code has absolute distance  $\ell' - (\ell'n/n')$ , we infer that any linear combination of the rows of the sub-matrix  $M''$  that yields the zero vector must yield zero in the column of  $M$  that corresponds to each of the heavy blocks (because a codeword with more than  $\ell'n/n'$  zeros must encode the zero of  $F$  (viewed as the all-zero sequence of  $\text{GF}(q')^\ell$ ). Recalling that there are  $n$  heavy blocks, it follows that an  $F'$ -linear combination of the rows of  $M''$  that yields the zero vector must yield zero in at least  $n$  columns of  $M$ . Using the second property of  $M$ , it follows that this  $F'$ -linear combination must contain more than  $n - m$  rows, and the claim follows. ◀

► **Construction 5.11** (a kind of condenser<sup>22</sup>). Let  $m, n', q', n''$  and  $M'$  be as in Claim 5.10, and suppose that  $q'$  is a power of two. Let  $\delta = 4n/n'$  and  $s = \Theta((q' \log n)^2/\delta)$ , and consider the following transformation of  $(z_{1,1}, \dots, z_{n'',s}) \in \{0, 1\}^{n''s}$  into an  $n$ -bit long string  $x = (x_1, \dots, x_n)$ .

1. For each  $i \in [n'']$ , compute  $z_i = \sum_{j \in [s]} z_{i,j} \text{ mod } q'$ .  
Viewing each  $z_i$  as an element of  $\text{GF}(q') \equiv \mathbb{Z}_{q'}$ , let  $z = (z_1, \dots, z_{n''}) \in \text{GF}(q')^{n''}$ .
2. Compute  $(y_1, \dots, y_n) = M'z \in \text{GF}(q')^n$ . For each  $i \in [n]$ , let  $x_i$  be the result of applying some balanced predicate to  $y_i$  (e.g.,  $x_i$  is the least significant bit of  $y_i$ ).<sup>23</sup>

Note that these computation can be carried out by constant-depth circuits of size  $\text{poly}(n)$ , since  $q' = \text{poly}(\log n)$ ,  $n' = \tilde{O}(n)$  and each row of  $M'$  has at most  $\text{poly}(\log n)$  non-zero entries.

► **Claim 5.12** (analysis of Construction 5.11). If the input to Construction 5.11 is taken from an  $(n''s, \delta n''s)$ -bit-fixing source, then there exist  $n - m$  bits in the output with a joint distribution that is  $n^{-\omega(1)}$ -close to  $U_{n-m}$ .

**Proof.** If a  $\delta$  fraction of the input bits are random, then for at least a  $\delta/2$  fraction of  $i \in [n'']$ , called good, at least a  $\delta/2$  fraction of the bits  $z_{i,1}, \dots, z_{i,s}$  are random. (Recall that random bits are independent of one another, whereas the other bits are fixed.) As shown by Kamp and Zuckerman [41], the sum modulo  $q'$  of the bits of a  $(s, \delta's)$ -bit-fixing source is  $\exp(-\Omega(\delta's/(q')^2))$ -close to the uniform distribution on  $\mathbb{Z}_{q'}$ . Hence, for each good  $i$  it holds that  $z_i$  is  $\exp(-\Omega(\delta s/(q')^2))$ -close to the uniform distribution on  $\mathbb{Z}_{q'}$  (and the  $z_i$ 's are independent of one another). By the choice of  $s = \Omega(q' \log n)^2/\delta$ , it follows that  $z_I$  is  $\exp(-\Omega(\log^2 n))$ -close to be uniform over  $\text{GF}(q')^{|I|}$ , where  $I$  denotes the set of good  $i$ 's and  $|I| \geq \delta n''/2$ .

Consider the column of  $M'$  that correspond to the good  $i$ 's. By the second property of  $M'$  (and using  $\delta = 4n/n'$ ), it holds that the rank of the corresponding  $n$ -by- $(2n/n') \cdot n''$  sub-matrix is at least  $n - m$ . Denoting a set of  $n - m$  linearly independent rows by  $R$ , it follows that  $y_R$  is  $\exp(-\Omega(\log^2 n))$ -close to be uniform over  $\text{GF}(q')^{n-m}$ , and a corresponding statement holds for  $x_R$  with respect to  $\{0, 1\}^{n-m}$ . The claim follows. ◀

<sup>22</sup> For any  $\delta \geq 1/\text{poly}(\log n)$  and any  $\rho \geq 1/\text{poly}(\log n)$ , this construction “condenses” an  $(n''s, \delta n''s)$ -bit-fixing source, into a *non-oblivious*  $(n, (1 - \rho) \cdot n)$ -bit-fixing source. Thus, the min-entropy rate is significantly increased (from  $\delta$  to  $1 - \rho$ ), but the output source belong to a wider class of sources.

<sup>23</sup> For this reason we need  $q'$  to be even.

**Wrapping up.** The final construction is as follows. Let  $C : \{0, 1\}^n \rightarrow \{0, 1\}$  be the  $\mathcal{AC}^0$  circuit guaranteed by Theorem 5.7. Setting  $\rho = \epsilon/O(\log n)^2 \geq 1/\text{poly}(\log n)$  and  $\delta = k(n^2)/n^2 \geq 1/\text{poly}(\log n)$ , we use  $n'' = 4n/\delta = \tilde{O}(n)$  and invoke Construction 5.11. Let  $C'' : \{0, 1\}^{n''s} \rightarrow \{0, 1\}^n$  be the  $\mathcal{AC}^0$  circuit provided by Construction 5.11, and note that  $\delta \leq k(n''s)/n''s$  (since we may assume that  $k(t)/t$  is non-increasing in  $t$ ). We define the final circuit  $C' : \{0, 1\}^{n''s} \rightarrow \{0, 1\}$  as the composition of  $C$  and  $C''$ ; that is,  $C'(z) = C(C''(z))$ . Using Claim 5.12, we infer that for any  $(n''s, \delta n''s)$ -bit-fixing source  $Z$ , it holds that  $C''(Z)$  is  $n^{-\omega(1)}$ -close to a source in which  $n - \rho n$  of the bits are totally random (and the rest may be determined as a function of them). By Theorem 5.7, in this case  $C(C''(Z))$  is  $\epsilon$ -close to a random bit.<sup>24</sup> Hence,  $C'$  is an  $\epsilon$ -error extractor for  $(n''s, k(n''s))$ -bit-fixing sources, which establishes the claim of the theorem. ◀

► **Remark (Construction 5.11, revisited).** While the output of Construction 5.11 is *not* an affine combination of its input bits, it is  $n^{-\omega(1)}$ -close to an affine source of min-entropy at least  $n - m$  (cf. [25]). This is due to the following two facts:

1. The vector  $(z_1, \dots, z_{n''}) \in (\{0, 1\}^{\ell''})^{n''}$  produced in Step 1 is  $n^{-\omega(1)}$ -close to a  $(n'', \delta n''/2, \ell'')$ -block-fixing source, where  $2^{\ell''} = q'$ .
2. The vector  $x$  is a GF(2)-linear combination of the bits of  $z$ , since  $y$  is a GF( $2^{\ell''}$ )-linear combination of the blocks of  $z$  (viewed as elements of GF( $2^{\ell''}$ )).

Hence, the bits of  $x$  are affine combinations of the non-fixed bits of  $z'$ , where  $z'$  is the  $(n'', \delta n''/2, \ell'')$ -block-fixing source that is  $n^{-\omega(1)}$ -close to  $(z_1, \dots, z_{n''})$ . Since the affine transformation of  $z'$  to  $x$  has rank at least  $n - m$  in the non-fixed variables of  $z'$ , our claim follows.

We now improve the construction asserted by Theorem 5.8 in two ways. First we show that the error of the extraction can be reduced (to a negligible in  $n$  level), and then we show that  $\text{poly}(\log n)$  bits can be extracted (rather than a single one). We start by observing that XORing values extracted from disjoint portions of a bit-fixing source yields an extractor of smaller error (alas it is guaranteed to work only for bit-fixing sources of a larger amount of min-entropy).<sup>25</sup>

► **Theorem 5.13** (error reduction for deterministic extraction from bit-fixing sources). *Suppose that  $E : \{0, 1\}^n \rightarrow \{0, 1\}$  is an  $\epsilon$ -error extractor for  $(n, k)$ -bit-fixing sources. Then, for every  $t \in \mathbb{N}$ , the function  $E' : \{0, 1\}^{tn} \rightarrow \{0, 1\}$ , given by  $E'(x_1, \dots, x_t) = \bigoplus_{i \in [t]} E(x_i)$  is an  $\epsilon^{\lceil tk/n \rceil}$ -error extractor for  $(tn, 2tk)$ -bit-fixing sources.*

Indeed, as detailed in Corollary 5.14 below, applying Theorem 5.13 to Theorem 5.8 yields a similar deterministic  $\mathcal{AC}^0$  extractor but for error rates that are smaller than  $1/\text{poly}(n)$ .

**Proof.** Letting  $\delta = k/n$ , we note that the existence of  $2\delta tn$  random bits in the source  $(X_1, \dots, X_t) \in (\{0, 1\}^n)^t$  implies that for at least a  $\delta$  fraction of the indices  $i \in [t]$  the source  $X_i$  is a  $(n, \delta n)$ -bit-fixing source (whereas the  $t$  sources are independent of one another). Since each of these  $\lceil \delta t \rceil$  extractions yields a bit that is  $\epsilon$ -close to uniform (whereas the  $t$  bits are independent of one another), the claim follows. ◀

► **Corollary 5.14** (improved error in deterministic  $\mathcal{AC}^0$ -extractors for bit-fixing sources). *For every  $k(n) \geq n/\text{poly}(\log n)$  and every  $\epsilon(n) \geq \exp(-\text{poly}(\log n))$ , there exist deterministic*

<sup>24</sup> Recall that  $\epsilon(n) \geq 1/\text{poly}(\log n)$ , which is much larger than the  $n^{-\omega(1)}$  deviation created by  $C''$ .

<sup>25</sup> Indeed, there is a trade-off, which we do not elaborate, between the amount of error reduction and the increase in the required min-entropy.

$\epsilon$ -error extractors  $E : \{0, 1\}^n \rightarrow \{0, 1\}$  for  $(n, k)$ -bit-fixing sources such that the extractors are computable in  $\mathcal{AC}^0$ .

**Proof.** Let  $E_0 : \{0, 1\}^{n_0} \rightarrow \{0, 1\}$  be the 0.1-error extractor for  $(n_0, k(k_0)/2)$ -bit-fixing sources provided by Theorem 5.8. Letting  $t = \Theta((n_0/k(n_0)) \log(1/\epsilon(n_0))) = \text{poly}(\log n_0)$  and  $n = t \cdot n_0$ , and applying Theorem 5.13, the claim follows, since  $t$ -wise XOR can be computed by constant-depth circuits of  $\text{poly}(n)$ -size. ◀

### Extracting slightly more bits

Applying the same idea as underlying the proof of Theorem 5.13, we can extract poly-logarithmically many bits rather than one.

▶ **Theorem 5.15** (Corollary 5.14, revisited). *For every  $k(n) \geq n/\text{poly}(\log n)$  and every  $\epsilon(n) \geq \exp(-\text{poly}(\log n))$ , there exist deterministic  $\epsilon$ -error extractors  $E : \{0, 1\}^n \rightarrow \{0, 1\}^{\text{poly}(\log n)}$  for  $(n, k)$ -bit-fixing sources such that the extractors are computable in  $\mathcal{AC}^0$ .*

**Proof.** We proceed in two steps, first obtaining an extractor that outputs double-logarithmically many bits, and next using it to establish the claim of the theorem. As in the proof of Corollary 5.14 (and other results), we shall use the fact that  $t$ -wise sums can be computed by constant-depth circuits of  $\exp(t^c)$ -size, for any constant  $c > 0$ .

▶ **Claim 5.16** (Corollary 5.14, revisited). *For every  $k(n) \geq n/\text{poly}(\log n)$ ,  $\epsilon(n) \geq \exp(-\text{poly}(\log n))$ , and  $\ell_1(n) = O(\log \log n)$ , there exist deterministic  $\epsilon$ -error extractors  $E : \{0, 1\}^n \rightarrow \{0, 1\}^{\ell_1(n)}$  for  $(n, k)$ -bit-fixing sources such that the extractors are computable in  $\mathcal{AC}^0$ .*

**Proof.** Letting  $\delta = k(n^2)/n^2$  and  $t = 2^{2\ell_1} \delta^{-1} \log(1/\epsilon(n^2)) = \text{poly}(\log n)$ , we note that the existence of  $\delta tn$  random bits in the source  $(X_1, \dots, X_t) \in (\{0, 1\}^n)^t$  implies that for at least a  $\delta/2$  fraction of the indices  $i \in [t]$  the source  $X_i$  is a  $(n, \delta n/2)$ -bit-fixing source (whereas the  $t$  sources are independent of one another). Applying the extractor of Corollary 5.14 to each of the  $t$  sources, we obtain a  $(t, \delta t/2)$ -bit-fixing source (or rather a  $t$ -bit string that is  $\exp(-\text{poly}(\log n))$ -close to such a source). Computing the sum of these  $t$  outputs modulo  $2^{\ell_1}$  (and invoking again the result of Kamp and Zuckerman [41]), we obtain a value that is  $\exp(-\Omega(\delta t/2^{2\ell_1}))$ -close (i.e.,  $\epsilon/2$ -close) to uniform over  $\mathbb{Z}_{2^{\ell_1}}$ . This yields an  $\epsilon$ -error extractor of  $\ell_1$  bits for  $(tn, \delta tn)$ -bit-fixing sources, and the claim follows. ◀

**Extracting  $\text{poly}(\log n)$  many bits.** For any  $\ell(n) = \text{poly}(\log n)$ , we will use an 0.1-biased sample space  $S \subset \{0, 1\}^\ell$  of size  $\text{poly}(\ell)$  (cf., e.g., [27, Sec. 8.5.2]), and let  $\ell_1 = \log |S| = O(\log \log n)$ . For any  $\epsilon(n) \geq \exp(-\text{poly}(\log n))$ , and  $k(n) \geq n/\text{poly}(\log n)$ , we again let  $\delta = k/n$ . This time we set  $t = \Theta(\ell + \log(1/\epsilon))/\delta = \text{poly}(\log n)$ , and consider a  $(tn, \delta tn)$ -bit-fixing source, denoted  $(X_1, \dots, X_t) \in (\{0, 1\}^n)^t$ , inferring that at least  $\delta/2$  fraction of the  $n$ -bit long  $X_i$ 's are  $(n, \delta n/2)$ -bit-fixing sources. Applying the extractor of Claim 5.16 to each of these  $t$  sources, we obtain  $t$  strings, each of length  $\ell_1$ , such that at least  $\delta t/2$  of these strings are  $\epsilon$ -close to being uniformly distributed in  $\{0, 1\}^{\ell_1}$ . In other words, we obtain a  $(t, \delta t/2, \ell_1)$ -block fixing source. Denoting the  $i^{\text{th}}$  block in this source by  $y_i$ , and viewing it as an element of  $[2^{\ell_1}] \equiv \{0, 1\}^{\ell_1}$ , we just output  $\oplus_{i \in [t]} s_{y_i}$ , where  $s_j$  is the  $j^{\text{th}}$  string in  $S$ . This  $\ell$ -bit output is  $0.1^{\delta t/2}$ -biased, since for any  $\alpha \in \{0, 1\}^\ell$  it holds that  $\langle \alpha, \oplus_{i \in [t]} s_{y_i} \rangle_2 = \oplus_{i \in [t]} \langle \alpha, s_{y_i} \rangle_2$ , where  $\langle \cdot, \cdot \rangle_2$  denotes inner product mod 2. It follows that the output is  $(2^{\ell/2} \cdot 0.1^{\delta t/2})$ -close to the uniform distribution over  $\{0, 1\}^\ell$ , and the theorem follows (using sufficiently large  $t = O(\ell + \log(1/\epsilon))/\delta$ ). ◀

### Extracting many more bits

Using the composition theorem of Gabizon *et al.* [26, Thm. 7.1], we obtain deterministic extractors that are computable in  $\mathcal{AC}^0$  and extract  $n/\text{poly}(\log n)$  bits. The aforementioned composition result requires three ingredients: (1) an adequate deterministic extractor (provided by Theorem 5.15), (2) an adequate seeded extractor (provided by Theorem 5.2), and (3) an adequate averaging sampler (provided by Theorem 3.8). We shall rely on the fact that the composition theorem of [26, Thm. 7.1] preserves  $\mathcal{AC}^0$  complexity.

► **Theorem 5.17** (deterministic  $\mathcal{AC}^0$  extraction of  $n/\text{poly}(\log n)$  bits from bit-fixing sources). *For every  $k(n) \geq n/\text{poly}(\log n)$  and every  $\epsilon(n) \geq 1/\text{poly}(n)$ , there exist deterministic  $\epsilon$ -error extractors  $E : \{0, 1\}^n \rightarrow \{0, 1\}^{n/\text{poly}(\log n)}$  for  $(n, k)$ -bit-fixing sources such that the extractors are computable in  $\mathcal{AC}^0$ .*

**Proof.** We start by reviewing the composition theorem of Gabizon *et al.* [26, Thm. 7.1], which is pivotal to our proof. The composition theorem of Gabizon *et al.* [26, Thm. 7.1] requires three ingredients (for some parameters  $\mu, \mu', \epsilon', \epsilon''$  etc):<sup>26</sup>

1. A deterministic  $\epsilon'$ -error extractor  $E' : \{0, 1\}^n \rightarrow \{0, 1\}^{r+r''}$  for  $(n, \mu't)$ -bit-fixing sources;
2. A seeded  $\epsilon''$ -error extractor  $E'' : \{0, 1\}^n \times \{0, 1\}^{r''} \rightarrow \{0, 1\}^m$  for  $(n, \mu n - t)$ -bit-fixing sources;
3. An  $(\mu, \mu', \gamma)$ -averaging sampler  $S : \{0, 1\}^r \rightarrow [n]^t$ .

It yields a deterministic  $\epsilon$ -error extractor  $E : \{0, 1\}^n \rightarrow \{0, 1\}^m$  for  $(n, \mu n)$ -bit-fixing sources and  $\epsilon = \epsilon'' + 2^{r+3} \cdot \epsilon' + 3\gamma$  that operates as follows. Denoting the first  $r$  (resp., last  $r''$ ) bits of  $E'$  by  $E'_1(x)$  (resp.,  $E'_2(x)$ ), it holds that  $E(x) = E''(x_{[n] \setminus S(E'_1(x))} \circ 0^t, E'_2(x))$ . Hence, if each of the three ingredients is computable by constant-depth  $\text{poly}(n)$ -size circuits, then  $E$  is in  $\mathcal{AC}^0$ .

Setting  $\mu = k(n)/n = 1/\text{poly}(\log n)$ , we shall use  $\mu' = \mu/2$  and  $t = k(n)/2$  so that  $\mu't = n/\text{poly}(\log n)$  and  $\mu n - t = n/\text{poly}(\log n)$ . Furthermore, we shall use  $\epsilon' = 2^{-\log^3 n}$ ,  $\epsilon'' = \gamma = 1/\text{poly}(n)$ ,  $m = n/\text{poly}(\log n)$ ,  $r'' = O(\log n)$  and  $r = O(\log n)^2$ . The required deterministic extractor is provided by Theorem 5.15, the seeded extractor is provided by Theorem 5.2, and the required averaging sampler is provided by Theorem 3.8. Hence we obtain  $\mathcal{AC}^0$  circuits computing  $E : \{0, 1\}^n \rightarrow \{0, 1\}^m$ , which is a deterministic  $\epsilon$ -error extractor for  $\epsilon = 2^{r+3} \cdot \epsilon' + 1/\text{poly}(n) = 1/\text{poly}(n)$ . ◀

### An explicit disperser

Recall that a deterministic disperser for a class of sources is a function that defines an onto mapping from the support of each source in the class to the range of the function. That is,  $D : \{0, 1\}^n \rightarrow \{0, 1\}^m$  is a deterministic disperser for a class of sources if for every source  $X$  in the class and over every  $v \in \{0, 1\}^m$  it holds that  $\Pr[D(X)=v] > 0$ .

<sup>26</sup> Actually, we present a special case of [26, Thm. 7.1], whereas the original version refers to a generalization of the notion of an averaging sampler. Loosely speaking, a  $(\mu, \mu', \mu'', \gamma)$ -averaging sampler is defined as in Definition 2.6, except that it refers to functions  $f : [n] \rightarrow [0, 1]$  such that  $\rho(f) = \mu$  and also requires that  $\Pr[\sum_{i \in S(U_r)} f(i) > t \cdot \mu''] \leq \gamma$ . Indeed, such an  $(\mu, \mu', 1, \gamma)$ -averaging sampler is an  $(\mu, \mu', \gamma)$ -averaging sampler as in Definition 2.6. We comment that using the original form of [26, Thm. 7.1] allows to obtain somewhat better parameters, by showing that a small variant on the construction establishing Theorem 3.8 yields the required  $(\mu, \mu', \mu'', \gamma)$ -averaging sampler. (Specifically, we shall set the parameters of the pairwise independent sampler, used in the latter proof, so that its error probability is  $o(\mu)$  rather than a constant, and establish the assertion for  $\mu'' = 2\ell \cdot \mu$  and  $\gamma = 1/\text{poly}(n)$ , where  $\ell = O(\log(1/\gamma))$ .)

► **Theorem 5.18** (deterministic disperser in  $\mathcal{AC}^0$  for bit-fixing sources). *For every  $k(n) \geq n/\text{poly}(\log n)$ , there exist explicit  $\mathcal{AC}^0$  circuits  $D : \{0, 1\}^n \rightarrow \{0, 1\}$  that constitute a disperser for  $(n, k)$ -bit-fixing sources.*

**Proof.** The first observation is that the Hamming weight of the outcome of any  $(n, k)$ -bit-fixing source is spread (unevenly) over an interval of  $k$  values. Hence, if we partition  $[n]$  into consecutive intervals of length  $k/10$ , then at least ten of these intervals will be assigned non-zero weight.<sup>27</sup> Our disperser outputs the least significant bit of the index of the interval in which the source’s outcome resides. The second observation is that for outcomes that reside in the middle portion of the interval we can determine the relevant index by approximate counting. Details follow.

Let  $\rho = k/n = 1/\text{poly}(\log n)$  and  $\epsilon = \rho/50$ , and recall that Ajtai [2] (see also [59]) provided explicit  $\mathcal{AC}^0$  circuits that compute the Hamming weight of  $n$ -bit strings up to an additive deviation of  $\epsilon n$ . Denoting this circuit by  $C$ , consider the circuit  $C'(x)$  that computes  $\lfloor C(x)/\epsilon n \rfloor \in \{0, 1, \dots, 1/\epsilon\}$ . Then,  $C'(x) \in \lfloor \text{wt}(x)/\epsilon n \rfloor \pm 2$ , where  $\text{wt}(x) \stackrel{\text{def}}{=} \sum_{i \in [n]} x_i$ . Hence, for every  $v \in [10/\rho]$  it holds that

$$\sum_{i=5v}^{5v+4} \Pr[C'(X) = i] \geq \Pr[\lfloor \text{wt}(X)/\epsilon n \rfloor = 5v + 2],$$

which means that if the latter term is positive then so is the former. Our disperser  $D$  outputs the least significant bit of  $\lfloor C'(x)/5 \rfloor$ .

For an arbitrary  $(n, k)$ -bit fixing source  $X$ , let  $u = \lfloor 50\mathbf{E}[\text{wt}(X)]/k \rfloor$ . Then, for every  $u' \in [u \pm 10]$ , it holds that  $\Pr[\lfloor 50\text{wt}(X)/k \rfloor = u'] > 0$ . It follows that  $\Pr[\lfloor C'(X)/5 \rfloor = \lfloor u/5 \rfloor] > 0$  and  $\Pr[\lfloor C'(X)/5 \rfloor = \lfloor u/5 \rfloor \pm 1] > 0$ , whereas in these two cases  $D$  outputs different values (since the least significant bit of  $\lfloor u/5 \rfloor$  is different from the least significant bit of  $\lfloor u/5 \rfloor \pm 1$ ). ◀

## 5.4 Extraction from zero-fixing sources

The impossibility results regarding bit-fixing sources (presented in Section 5.2) do not hold for a restricted class of such sources that was recently introduced by Cohen and Shinkar [19]. This class, called zero-fixing sources, consists of bit-fixing sources in which all fixed bits are set to zero.

► **Definition 5.19** (zero-fixing sources [19]). An  $(n, k)$ -zero-fixing source is a sequence of random variables  $X = (X_1, \dots, X_n) \in \{0, 1\}^n$  such that there exists a set of at least  $k$  indices  $I \subseteq [n]$  such that  $X_I$  is uniformly distributed over  $\{0, 1\}^{|I|}$  and  $X_i = 0$  for every  $i \in [n] \setminus I$ .

As shown next, there exist strong  $\mathcal{AC}^0$ -extractors for zero-fixing sources of *logarithmic min-entropy*, which was shown to be impossible for bit-fixing sources (see Theorem 5.4).

► **Theorem 5.20** (seeded  $\mathcal{AC}^0$ -extractor for zero-fixing sources). *Let  $k, m : \mathbb{N} \rightarrow \mathbb{N}$  and  $\epsilon : \mathbb{N} \rightarrow [0, 1]$  be such that  $m(n) = O(\log n)$  and  $k(n) = 2m(n) + O(\log(1/\epsilon)) \leq \text{poly}(\log n)$ . Then, there exists a function  $E : \{0, 1\}^n \times \{0, 1\}^{O(\log n)} \rightarrow \{0, 1\}^{m(n)}$  that is computable in uniform  $\mathcal{AC}^0$  and constitutes a strong  $\epsilon$ -error extractor for  $(n, k)$ -zero-fixing sources.*

<sup>27</sup> Indeed, it is likely that almost all the probability mass is concentrated in one interval (since the probability mass is concentrated in an interval of length  $o(k^{2/3})$ , but this does not contradict the foregoing.

**Proof.** On input  $x \in \{0, 1\}^n$  and a seed  $s \in \{0, 1\}^{O(\log n)}$ , the proposed extractor operates in two stages. In the first stage, which is deterministic, the extractor determines the first  $k = k(n)$  locations that are assigned the value 1 in  $x' = (x'_1, \dots, x'_{n+k}) \stackrel{\text{def}}{=} x1^k$ ; that is, it determines  $i_1 < \dots < i_k \leq n+k$  such that the  $i_k$ -bit long prefix of  $x'$  equals  $0^{i_1-1}10^{i_2-i_1-1}1 \cdot 0^{i_k-i_{k-1}-1}1$ . In other words, for each  $j \in [k]$ , the extractor determines  $i_j$  as the smallest  $i$  such that  $\sum_{\ell \in [i]} x'_\ell = j$ . (This is done using counters that count up to  $k+1 \leq \text{poly}(\log n)$ , while applying such counters to the strings  $x'_1 \dots x'_i$ , for  $i \in [n+k]$ .)

In the second stage, which uses the seed  $s$ , we apply a  $(k, \epsilon)$ -extractor to the sequence  $(i_1, \dots, i_k)$ , which is viewed as a string of length  $k \log n$ . Here, again, we can use the extractor of [29, Sec. 5], while relying on the fact that  $k \log n = \text{poly}(\log n)$ .

Note that if  $X$  is a  $(n, k)$ -zero-fixing extractor, then the sequence determined by the first stage has min-entropy at least  $k$ . To verify the claim, consider the set  $I$  of the first  $k$  non-fixed (i.e., random) bits in  $X$ . Then, the random set  $\{i \in I : X_i = 1\}$  has min-entropy  $k$ , since each of the  $2^k$  possible subsets is equally likely.  $\blacktriangleleft$

### Deterministic extraction

While our deterministic  $\mathcal{AC}^0$ -extractors for bit-fixing sources are not explicit (see Section 5.3), we show a very simple explicit construction for the case of zero-fixing sources (of comparable min-entropy rate). In fact, we prove a stronger result.

► **Theorem 5.21** (deterministic  $\mathcal{AC}^0$ -extractors for zero-fixing sources).

1. Let  $k : \mathbb{N} \rightarrow \mathbb{N}$  and  $\epsilon : \mathbb{N} \rightarrow [0, 1]$  be such that  $k(n) \geq n/\text{poly}(\log n)$  and  $\epsilon(n) \geq 1/\text{poly}(n)$ . Then, there exists a function  $E : \{0, 1\}^n \rightarrow \{0, 1\}^{n/\text{poly}(\log n)}$  that is computable in uniform  $\mathcal{AC}^0$  and constitutes an  $\epsilon$ -error extractor for  $(n, k)$ -zero-fixing sources.
2. Let  $k : \mathbb{N} \rightarrow \mathbb{N}$  and  $\epsilon : \mathbb{N} \rightarrow [0, 1]$  be such that  $k(n) = \Theta(\epsilon(n)^{-3} \log n) = \text{poly}(\log n)$ . Then, there exists a function  $E : \{0, 1\}^n \rightarrow \{0, 1\}$  that is computable in uniform  $\mathcal{AC}^0$  and constitutes an  $\epsilon$ -error extractor for  $(n, k)$ -zero-fixing sources.

**Proof Sketch.** Starting with Part 1 and letting  $\delta = k(n)/n \geq 1/\text{poly}(\log n)$ , we first present a simple extractor that outputs a single bit (and later apply the transformations underlying the proofs of Theorems 5.15 and 5.17). We consider a partition of the source into consecutive  $3t/\delta$ -bit long blocks, where  $t = \Theta(\epsilon_0^{-3} \log n)$  for any desired constant  $\epsilon_0 > 0$ . The extractor picks the first block that contains at least  $t$  ones, and outputs the parity of the bits in that block. (Recall that the block length is  $3t/\delta = \text{poly}(\log n)$ .)

Towards the analysis, we fix an arbitrary  $(n, k)$ -zero-fixing source, and let  $k_i$  denote the number of random bits in the  $i^{\text{th}}$  block, where  $i = 1, \dots, \delta n/3t$ . We consider corresponding random variables,  $Y_1, \dots, Y_{\delta n/3t}$ , such that  $Y_i$  denotes the sum of the bits in the  $i^{\text{th}}$  block. Clearly, there exists a block  $i$  such that  $k_i \geq 3t$ . Note that if  $k_i < t$ , then the  $i^{\text{th}}$  block will never be selected. On the other hand, if  $k_i \geq t$ , then, with probability at least  $1 - o(1/n)$ , it holds that  $Y_i = (0.5 \pm \epsilon_0) \cdot k_i$ . Hence, with probability at least  $1 - o(1)$ , some block  $i$  is selected, and it holds that  $k_i \geq (2 - 2\epsilon_0) \cdot t$ . Furthermore, for such a block  $i$  (i.e.,  $k_i \geq 2t - 2\epsilon_0 t$ ), the parity of the bits in the block (i.e.,  $Y_i \bmod 2$ ) has bias  $O(\epsilon_0)$ , also when conditioned on  $Y_i \geq t$ . To verify the last claim consider a pairing of the odd-parity strings of length  $k_i \geq 2t - 2\epsilon_0 t$  with the even-parity strings obtained by flipping the last bit. Now omit the strings that have Hamming weight smaller than  $t$ , and note that only an  $O(\epsilon_0)$  fraction of the remaining strings are left unmatched (since this omission may leave unmatched only strings of Hamming weight  $t$ , whereas  $\Pr[Y_i = t] = (1 + O(\epsilon_0))^j \cdot \Pr[Y_i = t + j]$  for every  $j \in [\epsilon_0^{-1}]$ ).

Part 1 follows by applying the transformations that underlie the proofs of Theorems 5.13, 5.15 and 5.17. Each of these transformations improves some parameter (i.e., error or output



length) of deterministic  $\mathcal{AC}^0$ -extractors for bit-fixing sources of min-entropy  $n/\text{poly}(\log n)$ . The point is that these transformation only use the fact that certain sub-sources derived from the given bit-fixing source are bit-fixing sources with certain parameters (where sub-sources are projections of the source on some locations).<sup>28</sup>

The same holds for zero-fixing sources.

Turning to Part 2, we consider the following extractor, which, for every  $\ell = 0, 1, \dots, \log(n/k(n))$ , uses a family of  $n^2$  pairwise independent hashing functions  $h : [n] \rightarrow [2^\ell]$ . On input  $x \in \{0, 1\}^n$ , the extractor finds  $\ell$  such that for most  $h : [n] \rightarrow [2^\ell]$  it holds that  $S_h \stackrel{\text{def}}{=} \{i \in [n] : x_i = 1 \ \& \ h(i) = 1\}$  has cardinality at least  $(1 - \epsilon(n)) \cdot k(n)$  and at most  $(2 + \epsilon(n)) \cdot k(n)$ . It then picks the first of these (majority)  $h$ 's, and outputs the parity of  $|S_h|$ . (This is done by using a counter that counts till  $3k(n) = \text{poly}(\log n)$ ; note that hashing functions in these families can be computed by depth-two circuits of size  $\text{poly}(n)$ .)

Towards the analysis, we fix an arbitrary  $(n, k)$ -zero-fixing source  $X$ , and let  $I$  denote the set of random (i.e., non-fixed) bit locations in it. Then, there exists an  $\ell$  such that  $|I|$  is in  $[2^{\ell+1} \cdot k, 2^{\ell+2} \cdot k)$ , where  $k = k(n)$ , and a hashing function  $h : [n] \rightarrow [2^\ell]$  such that  $|\{i \in I : h(i) = 1\}| \in (2k - \epsilon k, 4k + \epsilon)$ , where  $\epsilon = \epsilon(n)$ . Using an analysis as in Part 1, we infer that with high probability the pair  $(\ell, h)$  chosen by the extractor satisfies  $|\{i \in I : h(i) = 1\}| \in (2k - 2\epsilon k, 4k + 2\epsilon k)$ , where here we use the fact that for the majority of the  $h$ 's it holds that  $|S_h| \in [(1 - \epsilon) \cdot k, (2 + \epsilon) \cdot k]$ . Similarly, we conclude that, for the selected  $h$ , the parity of  $|S_h|$  has bias  $O(\epsilon)$ , and Part 2 follows. ◀

► Remark (zero-block-fixing sources, generalizing Definition 5.19). An  $(n, k, \ell)$ -zero-block-fixing source is a sequence of random variables  $X = (X_1, \dots, X_n) \in \{0, 1\}^{n\ell}$  such that there exists a set of at least  $k$  indices  $I \subseteq [n]$  such that  $X_I$  is uniformly distributed over  $\{0, 1\}^{|I|\ell}$  and  $X_i = 0^\ell$  for every  $i \in [n] \setminus I$ . Extraction from such a source is quite easy, since each non-fixed block is clearly identified as such (and is uniformly distributed on  $\{0, 1\}^\ell \setminus \{0^\ell\}$ ).

## 6 Extraction with long seeds

In this section we present extractors that use very long seeds, which is indeed uncustomary in the studies of extractors, but the point is that these extractors establish the tightness of Theorem 5.5. Recall that (Part 1 of) of Theorem 5.5 establishes lower bounds on the relationship between the min-entropy  $k(n)$  and the seed-length  $r(n)$  for any non-trivial extractor computable in  $\mathcal{AC}^0$ . Specifically, it asserts that  $k(n) \cdot r(n) \geq n/\text{poly}(\log n)$  must hold (for any non-trivial extraction in  $\mathcal{AC}^0$ ). The point of the current section is showing that these bounds are tight. Specifically, we shall show that  $k(n) \cdot r(n) \geq n/\text{poly}(\log n)$  suffices for trivial extraction in  $\mathcal{AC}^0$ .

We shall first show (see Section 6.1) that  $\mathcal{AC}^0$  circuits can extract  $n + \text{poly}(\log n)$  bits using a seed of length  $n$ , provided that  $k(n) \geq \text{poly}(\log n)$ . Actually, we show that  $r(n) + \text{poly}(\log n)$  bits can be extracted in  $\mathcal{AC}^0$  provided that  $k(n) \cdot r(n) \geq n/\text{poly}(\log n)$  (and  $k(n) \geq \text{poly}(\log n)$ ). This is a special case of a more general result (presented in Section 6.2) that asserts that  $r(n) + m'(n)$  bits can be extracted in  $\mathcal{AC}^0$  provided that  $k(n) \cdot r(n) \geq m'(n) \cdot n/\text{poly}(\log n)$  (and  $m'(n) \leq k(n)/2$  and  $k(n) \geq \text{poly}(\log n)$ ). The latter result is obtained by combining extractors presented in previous part of this write-up (including the one presented in Section 6.1) with a simple scheme that amounts to repeated extraction with independent seeds.

<sup>28</sup>This holds also for [26, Thm. 7.1], which is used within the proof of Theorem 5.17.



### 6.1 Extraction with a seed of linear length

The following construction generalizes an  $\mathcal{AC}^0$ -computable extractor presented by Viola [58, Lem. 4.3], where the original construction corresponded to the case  $\ell = 1$ .

► **Theorem 6.1** (the inner product extractor). *Let  $n, \ell \in \mathbb{N}$  such that  $\ell/2$  is a power of 3 and  $\ell \leq \text{poly}(\log n)$ . Then, there exists a  $(\ell + 3 \log(2/\epsilon), \epsilon)$ -extractor,  $E : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}^{n+\ell}$ , computable by uniform  $\mathcal{AC}^0$ .*

The requirement that  $\ell$  be of a special form (i.e.,  $\ell/2$  is a power of 3) can be dropped when  $\ell = O(\log n)$ . This special form is only used for asserting the existence of (uniform) circuits of constant depth and size  $\text{poly}(n)$  for computing inverses in  $\text{GF}(2^\ell)$ .

**Proof.** Our starting point is the strong extractor  $E' : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}^\ell$  that views its input source  $x$  and seed  $r$  as  $n/\ell$ -long sequences over  $\text{GF}(2^\ell)$  and outputs their inner product, where the arithmetics is of the said field. That is,  $E'(x, r) = \sum_{i \in [t]} x_i r_i$ , where  $t = n/\ell$ ,  $x = (x_1, \dots, x_t)$  and  $r = (r_1, \dots, r_t)$ . Note that for a random  $r$ , the mapping  $x \mapsto E'(x, r)$  is pairwise independent, and so  $E'$  is a strong  $(\ell + 3 \log(2/\epsilon), \epsilon)$ -error extractor (by the Leftover Hashing Lemma, see, e.g., [27, Thm. D.4]). Hence,  $E''(x, r) = r \circ E'(x, r)$  is an  $(\ell + 3 \log(2/\epsilon), \epsilon)$ -error extractor with output length  $n + \ell$ .

Wishing to obtain  $\mathcal{AC}^0$  circuits, we define  $E(x, s) = E''(x, f(x, s)) = (f(x, s), E'(x, f(x, s)))$ , where  $f(x, s) = (f_1(x, s), \dots, f_t(x, s)) \in \text{GF}(2^\ell)^t$  is defined as follows.

- Fictitiously define  $s_0 = 0 \in \text{GF}(2^\ell)$  and  $x_0 = 1 \in \text{GF}(2^\ell)$ .
- Let  $\text{prv}_x(i) = j \in \{0, 1, \dots, i-1\}$  denote the index of the last non-zero element that precedes  $i$ ; that is,  $\text{prv}_x(i) = j \in \{0, 1, \dots, i-1\}$  if  $x_j \neq 0$  and  $x_{j+1} = \dots = x_{i-1} = 0$ .
- Finally, define  $f_i(x, s) = s_i$  if  $x_i = 0$  and  $f_i(x, s) = (s_i - s_{\text{prv}_x(i)})/x_i$  otherwise.

In particular,  $f(0^t, s) = s$  and  $f_i(1^t, s) = s_i - s_{i-1}$  (for every  $i \in [t]$ ). Note that for  $\ell = 1$ , we have  $f_i(x, s) = s_i$  if  $x_i = 0$  and  $f_i(x, s) = s_i - s_{\text{prv}_x(i)}$  otherwise.

Before showing that  $E$  is in  $\mathcal{AC}^0$ , we analyze its output distribution. Note that, for every fixed  $x$ , the mapping  $s \mapsto f(x, s)$  is a bijection, and it follows that  $f(x, U_n)$  is distributed identically to  $U_n$ . Hence,  $E(x, U_n) = (f(x, U_n), E'(x, f(x, U_n)))$  is distributed identically to  $E''(x, U_n) = (U_n, E'(x, U_n))$ , since in both distributions the last  $\ell$  bits are obtained by applying the same function (i.e.,  $E'(x, \cdot)$ ) to the first  $n$  bits. It follows that  $E''$  is an  $(\ell + 3 \log(2/\epsilon), \epsilon)$ -error extractor with output length  $n + \ell$ . It is left to show that  $E$  can be computed in uniform  $\mathcal{AC}^0$ .

Turning to the complexity of  $E$ , we first consider the computation of  $E'(x, f(x, s))$ . Note that  $E'(0^t, r) = 0 \in \text{GF}(2^\ell)$  for every  $r$ , whereas for  $x \neq 0^t$ , it holds that  $E'(x, f(x, s))$  equals

$$\begin{aligned} \sum_{i \in [t]} x_i f_i(x, s) &= \sum_{i: x_i \neq 0} x_i f_i(x, s) \\ &= \sum_{i: x_i \neq 0} x_i \cdot (s_i - s_{\text{prv}_x(i)})/x_i \\ &= \sum_{i: x_i \neq 0} s_i - \sum_{i: x_i \neq 0} s_{\text{prv}_x(i)} \end{aligned}$$

which equals  $s_i$  for the largest  $i$  such that  $x_i \neq 0$ . Hence,  $E'(x, f(x, s)) = \bigvee_{i \in [n]} \chi_i \cdot s_i$ , where  $\chi_i = ((x_i \neq 0) \wedge \bigwedge_{i < k \leq n} (x_k = 0))$  and  $\sigma \cdot (\tau_1, \dots, \tau_\ell) = (\sigma \wedge \tau_1, \dots, \sigma \wedge \tau_\ell)$  for every  $\sigma \in \{0, 1\}$  and  $(\tau_1, \dots, \tau_\ell) \in \{0, 1\}^\ell \equiv \text{GF}(2^\ell)$ .

Finally, we consider the computation of the  $f_i$ 's. We first note that  $\text{prv}_x(i)$  can be computed by uniform  $\mathcal{AC}^0$  (by computing the bits  $((x_j \neq 0) \wedge \bigwedge_{j < k < i} (x_k = 0))$  for  $j \in \{0, 1, \dots, i-1\}$ ). Next, we note that for  $\ell = \text{poly}(\log n)$ , addition and multiplication in

$\text{GF}(2^\ell)$  are computable by (uniform) constant-depth circuits of  $\text{poly}(n)$ -size.<sup>29</sup> We can take inverses in  $\text{GF}(2^\ell)$  by raising to the power  $2^\ell - 2$ , but the question is whether this can be done in by (uniform) circuits of constant depth and size  $\exp(\ell^c)$  for any desired  $c > 0$ . Fortunately, for  $\ell$  of the form  $2 \cdot 3^i$ , the answer is positive – see [33, Cor. 6 (2)]. ◀

**An alternative construction**

As an alternative generalization of the extractor presented by Viola [58, Lem. 4.3], consider using the seed (which will now have length  $2n$  rather than  $n$ ) as a description of a sequence of  $n' = n/\ell$  Toeplitz matrices, denoted  $T = (T_1, \dots, T_{n'})$ , where each  $T_i$  is a  $\ell$ -by- $\ell$  matrix.<sup>30</sup> Likewise, we view the source  $x$  as a sequence  $(x_1, \dots, x_{n'}) \in (\{0, 1\}^\ell)^{n'}$ , and output  $E'(x, T) = (T, Tx)$ , where  $Tx = \sum_{i \in [n']} T_i x_i$ . Recall that  $E'$  is a strong  $(\ell + 3 \log(2/\epsilon), \epsilon)$ -error extractor; again, this follows by the the Leftover Hashing Lemma (using the fact that  $x \mapsto E'(x, U_n)$  is pairwise independent).

It is simpler to implement this extractor in  $\mathcal{AC}^0$  when using affine transformations based on  $\ell$ -by- $\ell$  Toeplitz matrices; that is, rather than the  $T_i$ 's we use  $A_i = (T_i, b_i)$ 's, where  $b_i \in \{0, 1\}^\ell$ , and output the pair  $((A_1, \dots, A_{n'}), v)$  such that  $v = \sum_{i \in [n']} (T_i x_i + b_i)$ . That is, for  $A = (A_1, \dots, A_{n'})$ , where  $A_i = (T_i, b_i)$ , the extractor is defined by  $E(x, A) = (A, \sum_{i \in [n']} (T_i x_i + b_i))$ . Indeed,  $v = \sum_{i \in [n']} T_i x_i + \sum_{i \in [n']} b_i$ , but it is useful to have this redundancy. Specifically, the  $\mathcal{AC}^0$  implementation of  $E$  uses its seed to determine  $(A_1, \dots, A_{n'})$ , where  $A_i = (T_i, b_i)$ , but then outputs  $((T_1, b'_1), \dots, (T_{n'}, b'_{n'}), T_{n'} x_{n'} + b_{n'})$  such that  $b'_i = b_i + (T_{i-1} x_{i-1} + b_{i-1})$ , where  $T_0 x_0 + b_0 = 0^\ell$ . The key observation is that  $T_{n'} x_{n'} + b_{n'}$  equals the last  $\ell$  bits of  $E(x, ((T_1, b'_1), \dots, (T_{n'}, b'_{n'})))$ . This holds because

$$\begin{aligned} \sum_{i \in [n']} (T_i x_i + b'_i) &= \sum_{i \in [n']} (T_i x_i + b_i + T_{i-1} x_{i-1} + b_{i-1}) \\ &= T_{n'} x_{n'} + b_{n'}. \end{aligned}$$

Turning back to the construction that uses linear (rather than affine) transformations, we mention that it can be implemented by using the first column in each matrix  $T_i$  that corresponds to a 1-entry in  $x_i$ .

**Using a shorter seed when the min-entropy is  $\omega(\log^3 n)$**

Combining the extractor of Theorem 6.1 with a suitable averaging sampler, we obtain the following.

► **Corollary 6.2** (on the tightness of Theorem 5.5). *For any  $k = \Omega(\log n)$  and  $r \geq T \stackrel{\text{def}}{=} \min(n, \Theta(n \log^3 n)/k)$ , there exists a  $(k, \text{poly}(1/n))$ -extractor,  $E : \{0, 1\}^n \times \{0, 1\}^r \rightarrow \{0, 1\}^{r+\ell}$ , where  $\ell = \min(\Omega(k), \text{poly}(\log n))$ , that is computable by uniform  $\mathcal{AC}^0$ .*

We stress the fact that Corollary 6.2 implies an extractor that outputs  $m(n) = r(n) + \Omega(\log n) > r(n)$  bits using a seed of length  $r(n) = O(n \log^3 n)/k(n)$  (provided that  $k(n) = \Omega(\log n)$ ). Hence, non-trivial extraction in  $\mathcal{AC}^0$  (i.e.,  $m(n) > r(n)$ ) is possible whenever  $k(n) \cdot r(n) = O(n \log^3 n)$  (provided  $k(n) = \Omega(\log n)$ ). Recall that (Part 1 of) Theorem 5.5

<sup>29</sup> In particular, recall that multiplication in  $\text{GF}(2^\ell)$  reduces to computing inner products mod 2 of  $\ell$ -bit long vectors, whereas such computation can be carried out by depth  $d = O(1)$  circuits of size  $\exp(\ell^{1/(d-1)}) = \text{poly}(n)$ .

<sup>30</sup> As usual, each Toeplitz matrix is represented by its first row and its first column. For notational simplicity we view strings as either row or column vectors, according to their use.

asserts that  $k(n) \cdot m(n) = \Omega(n/\text{poly}(\log n))$  must hold for any non-trivial extraction in  $\mathcal{AC}^0$ . Loosely speaking, the combination of these results implies that *non-trivial extraction in  $\mathcal{AC}^0$  is possible if and only if  $k(n) \cdot m(n) = \tilde{\Theta}(n)$ .*

**Proof.** (The case of  $T = n$  follows immediately from Theorem 6.1, and we focus on the case that  $T < 0.1n$  (since otherwise, we can artificially increase  $T$  to  $n$  and act accordingly).) Assuming that  $T \leq 0.1n$ , we combine an adequate averaging sampler with the extractor of Theorem 6.1, where the combination uses the sample-then-extract paradigm (as stated in Corollary 2.8). Actually, we plug the extractor of Theorem 6.1 into Corollary 3.9, which already incorporate the adequate sampler. Specifically, when invoking Corollary 3.9, we set  $r_0 = t = r - O(\log n)^2$  and  $\delta = k(n)/n$  (and use  $\epsilon = 1/\text{poly}(n)$  and (say)  $\beta = 1/2$ ).<sup>31</sup> ◀

## 6.2 Repeated extraction with independent seeds

The following straightforward scheme for repeated extraction is wasteful in its use of the seed, but its appeal lies in the fact that it preserves the computational complexity of the original extractor. The key observation underlying its analysis is that (typical) conditioning on  $m'$  bits extracted from an  $(n, k)$ -source yields an  $(n, k - m' - 1)$ -source; this observation is at least implicit in numerous works regarding randomness extraction.

► **Theorem 6.3** (repeated extraction). *Let  $n, k, r, m, t \in \mathbb{N}$  and  $\epsilon \in [0, 1]$ . If  $E : \{0, 1\}^n \times \{0, 1\}^r \rightarrow \{0, 1\}^m$  is a  $(k, \epsilon)$ -extractor, then  $E_t : \{0, 1\}^n \times \{0, 1\}^{tr} \rightarrow \{0, 1\}^{tm}$  defined by  $E_t(x, s_1 \circ \dots \circ s_t) = E(x, s_1) \circ \dots \circ E(x, s_t)$  is a  $(k + (t - 1) \cdot (m + 1), (2t - 1) \cdot \epsilon)$ -extractor. An analogous statement holds for strong extractors.*

(Hence, the quality of extraction is harmed in a small manner, especially when  $t \ll k/m$ . Indeed, the result is meaningful only if  $k + (t - 1)(m + 1) < n$ .)

**Proof.** We first prove the version that refers to ordinary extractors.<sup>32</sup> We proceed by induction on  $t$ , proving that for any  $(n, k + (t - 1)(m + 1))$ -source  $X$  the statistical distance between  $U_{tm}$  and  $E_t(X, U_{td})$  is at most  $(2t - 1) \cdot \epsilon$ . The base case (of  $t = 1$ ) is immediate by the hypothesis regarding  $E$ . In the induction step we proceed as follows, while writing  $E_t(x, s_1 \circ \dots \circ s_t) = E_{t-1}(x, s_1 \circ \dots \circ s_{t-1}) \circ E(x, s_t)$ :

$$\begin{aligned} & \Delta[E_t(X, U_{tr}); U_{tm}] \\ &= \Delta[E_{t-1}(X, U_{(t-1)r}) \circ E(X, U_r); U_{(t-1)m} \circ U_m] \\ &\leq \Delta[E_{t-1}(X, U_{(t-1)r}) \circ E(X, U_r); U_{(t-1)m} \circ E(X, U_r)] \\ &\quad + \Delta[U_{(t-1)m} \circ E(X, U_r); U_{(t-1)m} \circ U_m] \\ &\leq \Delta[E_{t-1}(X, U_{(t-1)r}) \circ E(X, U_r); U_{(t-1)m} \circ E(X, U_r)] + \Delta[E(X, U_r); U_m], \end{aligned} \quad (13)$$

where the last inequality uses  $\Delta[\Pi(Y); \Pi(Z)] \leq \Delta[Y; Z]$  (for any random process  $\Pi$ ). Using the hypothesis regarding  $E$ , the second term of Eq. (13) is upper bounded by  $\epsilon$ . So we turn to analyze the first term of Eq. (13). We shall use the following notation:

■ We let  $k' = k + (t - 1)(m + 1)$  and recall that  $X$  is an  $(n, k')$ -source;

<sup>31</sup> We mention that an essentially weaker statement, which refers only to  $k \geq n^{2/3}$ , follows by instantiating Corollary 3.5 with the extractor of Theorem 6.1. In this case, the resulting extractor has seed length  $r = t + O(\log n)$  rather than  $t + O(\log n)^2$ , but this gain is insignificant because in both cases  $t = \Omega(\log^3 n)$ .

<sup>32</sup> The proof of the version that refers to strong extraction is very similar, and will be presented later.

- We let  $Y$  denote the distribution of  $E(X, U_r)$ ;
- For any  $y$  in the support of  $Y$ , we let  $X'_y$  denote the distribution of  $X$  conditioned on  $E(X, U_r) = y$ .

Using these notations we have

$$\begin{aligned} & \Delta[E_{t-1}(X, U_{(t-1)r}) \circ E(X, U_r); U_{(t-1)m} \circ E(X, U_r)] \\ &= \mathbf{E}_{y \leftarrow Y}[\Delta[E_{t-1}(X'_y, U_{(t-1)r}); U_{(t-1)m}]] \\ &\leq \max_{y: \mathbf{H}_\infty(X'_y) \geq k' - m - 1} \{ \Delta[E_{t-1}(X'_y, U_r); U_{(t-1)m}] \} \tag{14} \\ &\quad + \Pr_{y \leftarrow Y}[\mathbf{H}_\infty(X'_y) < k' - m - 1], \tag{15} \end{aligned}$$

where the inequality uses the fact that  $\Delta[;]$  is upper bounded by 1. We upper bound Eq. (14) by using the induction hypothesis, while noting that in this case  $X'_y$  is an  $(n, k + (t - 1)(m + 1) - m - 1)$ -source. Hence, Eq. (14) is upper bounded by  $(2(t - 1) - 1) \cdot \epsilon$ . To bound Eq. (15), we first observe that

$$\Pr_{y \leftarrow Y} [\Pr[E(X, U_r) = y] < 0.5 \cdot 2^{-m}] < \epsilon, \tag{16}$$

because otherwise the hypothesis regarding  $E$  is violated. (Specifically, let  $B = \{y : \Pr[E(X, U_r) = y] < 0.5 \cdot 2^{-m}\}$ , then  $\Pr[U_m = y] = 2^{-m} > 2 \cdot \Pr[Y = y]$  for every  $y \in B$ , which implies  $\Pr[U_m \in B] > 2 \cdot \Pr[Y \in B]$ , and so  $\Pr[Y \in B] \geq \epsilon$  implies  $\Delta[Y; U_m] > \epsilon$ .) Now, using Eq. (16), it follows that

$$\begin{aligned} & \Pr_{y \leftarrow Y}[\mathbf{H}_\infty(X'_y) < k' - m - 1] \\ &\leq \Pr_{y \leftarrow Y}[\Pr[E(X, U_r) = y] < 2^{-m-1}] \\ &\quad + \Pr_{y \leftarrow Y}[\exists x \text{ s.t. } \Pr[X'_y = x] > 2^{-k'+m+1} \mid \Pr[E(X, U_r) = y] \geq 2^{-m-1}] \\ &\leq \epsilon + \Pr_{y \leftarrow Y}[\exists x \text{ s.t. } \Pr[X = x | E(X, U_r) = y] > 2^{-k'+m+1} \mid \Pr[E(X, u) = y] \geq 2^{-m} \tag{17}] \end{aligned}$$

Next, note that for every  $x \in \{0, 1\}^n$  and  $y \in \{0, 1\}^m$ , it holds that

$$\Pr[X'_y = x] = \Pr[X = x | E(X, U_r) = y] \leq \frac{\Pr[X = x]}{\Pr[E(X, U_r) = y]}$$

where equality holds if and only if  $\Pr[E(X, U_r) = y] = 1$ . Hence, for  $y$  such that  $\Pr[E(X, U_r) = y] \geq 2^{-m-1}$ , and for every  $x$ , it holds that

$$\begin{aligned} \Pr[X = x | E(X, U_r) = y] &\leq \frac{\Pr[X = x]}{\Pr[E(X, U_r) = y]} \\ &\leq 2^{-k'+m+1} \end{aligned}$$

which means that the second term in Eq. (17) is zero. Hence, Eq. (15) is upper bounded by  $\epsilon$ , and the induction claim follows, since Eq. (13) is upper bounded by  $\epsilon + (2(t - 1) - 1) \cdot \epsilon + \epsilon$ .

**The case of strong extractors.** We now turn to the version that refers to strong extractors, and proceed as in the ordinary case subject to adequate modifications. (Indeed, the reader may want to skip the rest of the proof.) Denoting  $E'_t(x, \bar{u}) = E_t(x, \bar{u}) \circ \bar{u}$  (and  $E'(x, u) =$

$E(x, u) \circ u$ ), we prove (by induction on  $t$ ) that for any  $(n, k + (t - 1)(m + 1))$ -source  $X$  the statistical distance between  $U_{tm+tr}$  and  $E'_t(X, U_{tr})$  is at most  $(2t - 1) \cdot \epsilon$ . Here we use:

$$\begin{aligned} & \Delta[E'_t(X, U_{tr})U_{tm+tr}] \\ &= \Delta[E'_{t-1}(X, U_{(t-1)r}) \circ E'(X, U_r); U_{(t-1)(m+r)} \circ U_{m+r}] \\ &\leq \Delta[E'_{t-1}(X, U_{(t-1)r}) \circ E'(X, U_r); U_{(t-1)(m+r)} \circ E'(X, U_r)] + \Delta[E'(X, U_r); U_{m+r}]. \end{aligned} \quad (18)$$

Using the (strong) hypothesis regarding  $E$ , the second term of Eq. (18) is upper bounded by  $\epsilon$ . So we turn to analyze the first term of Eq. (18). We shall use the following notation:

- For any  $u \in \{0, 1\}^r$ , we let  $Y_u$  denote the distribution of  $E(X, u)$ ;
- For any  $u \in \{0, 1\}^r$  and  $y$  in the support of  $Y_u$ , we let  $X'_{u,y}$  denote the distribution of  $X$  conditioned on  $E(X, u) = y$ .

Using these notations we have

$$\begin{aligned} & \Delta[E'_{t-1}(X, U_{(t-1)r}) \circ E'(X, U_r); U_{(t-1)(m+r)} \circ E'(X, U_r)] \\ &= \mathbf{Pr}_{u \leftarrow U_r; y \leftarrow Y_u} [\Delta[E'_{t-1}(X'_{u,y}, U_{(t-1)r}); U_{(t-1)(m+r)}]] \\ &\leq \max_{u, y: \mathbf{H}_\infty(X'_{u,y}) \geq k' - m - 1} \{ \Delta[E'_{t-1}(X'_{u,y}, U_r); U_{(t-1)(m+r)}] \} \end{aligned} \quad (19)$$

$$+ \mathbf{Pr}_{u \leftarrow U_r; y \leftarrow Y_u} [\mathbf{H}_\infty(X'_{u,y}) < k' - m - 1]. \quad (20)$$

We upper bound Eq. (19) by using the induction hypothesis, while noting that in this case  $X'_{u,y}$  is an  $(n, k + (t - 1)(m + 1) - m - 1)$ -source. Hence, Eq. (19) is upper bounded by  $(2(t - 1) - 1) \cdot \epsilon$ . To bound Eq. (20), we first observe that

$$\mathbf{Pr}_{u \leftarrow U_r; y \leftarrow Y_u} [\mathbf{Pr}[E(X, u) = y] < 0.5 \cdot 2^{-m}] < \epsilon \quad (21)$$

because otherwise the hypothesis regarding  $E$  is violated. (Specifically, let  $B = \{(u, y) : \mathbf{Pr}[E(X, u) = y] < 0.5 \cdot 2^{-m}\}$ , then  $\mathbf{Pr}[U_{r+m} \in B] > 2 \cdot \mathbf{Pr}[(U_r, Y_{U_r}) \in B]$ , since  $\mathbf{Pr}[U_m = y] > 2 \cdot \mathbf{Pr}[Y_u = y]$  for every  $(u, y) \in B$ .) It follows that

$$\begin{aligned} & \mathbf{Pr}_{u \leftarrow U_r; y \leftarrow Y_u} [\mathbf{H}_\infty(X'_{u,y}) < k' - m - 1] \\ &\leq \mathbf{Pr}_{u \leftarrow U_r; y \leftarrow Y_u} [\mathbf{Pr}[E(X, u) = y] < 2^{-m-1}] \end{aligned} \quad (22)$$

$$+ \mathbf{Pr}_{u \leftarrow U_r; y \leftarrow Y_u} [\exists x \text{ s.t. } \mathbf{Pr}[X'_{u,y} = x] > 2^{-k'+m+1} \mid \mathbf{Pr}[E(X, u) = y] \geq 2^{-m-1}], \quad (23)$$

where Eq. (22) is upper bounded by  $\epsilon$ . Next, note that for every  $x \in \{0, 1\}^n$ ,  $u \in \{0, 1\}^r$  and  $y \in \{0, 1\}^m$ , it holds that

$$\mathbf{Pr}[X'_{u,y} = x] = \mathbf{Pr}[X = x \mid E(X, u) = y] \leq \frac{\mathbf{Pr}[X = x]}{\mathbf{Pr}[E(X, u) = y]}$$

where equality holds if and only if  $E(x, u) = y$ . Hence, for  $u$  and  $y$  such that  $\mathbf{Pr}[E(X, u) = y] \geq 2^{-m-1}$ , and for every  $x$ , it holds that

$$\begin{aligned} \mathbf{Pr}[X = x \mid E(X, u) = y] &\leq \frac{\mathbf{Pr}[X = x]}{\mathbf{Pr}[E(X, u) = y]} \\ &\leq 2^{-k'+m+1} \end{aligned}$$

which means that the the second term in Eq. (23) is zero. Hence, Eq. (20) is upper bounded by  $\epsilon$ , and the induction claim follows (since Eq. (18) is upper bounded by  $\epsilon + (2(t-1)-1) \cdot \epsilon + \epsilon$ ). ◀

**Applications to the study of extraction in  $\mathcal{AC}^0$**

Combining Theorem 6.3 with the various extractors presented in this work, we obtain.

► **Corollary 6.4** (on extraction in  $\mathcal{AC}^0$  using a long seed). *For every  $\epsilon : \mathbb{N} \rightarrow (0, 1]$  such that  $\epsilon(n) \geq 1/\text{poly}(n)$  and every constant  $\alpha < 1$ , there exist explicit  $\mathcal{AC}^0$  circuits that compute  $(k, \epsilon)$ -extractors,  $E : \{0, 1\}^n \times \{0, 1\}^{r(n)} \rightarrow \{0, 1\}^{m(n)}$ , for the following relation between  $r, m$  and  $k$ .*

1. *For any  $k(n) = \Omega(n)$ , we can have any  $m(n) \leq \alpha \cdot k(n)$  with any  $r(n) = \max(\beta \cdot m(n), \Omega(\log n))$ , for any constant  $\beta > 0$ . Furthermore, the extractor is strong and the  $\mathcal{AC}^0$  circuits have depth three.*
2. *For any  $k(n) \geq n/\text{poly}(\log n)$ , we can have any  $m(n) \leq \alpha \cdot k(n)$  with  $r(n) = \max(\beta \cdot m(n), \Omega(\log n))$ , for any constant  $\beta > 0$ . Furthermore, the extractor is strong and the  $\mathcal{AC}^0$  circuits have depth  $4 + \frac{\log(n/k(n))}{\log \log n}$ .*
3. *For any constant  $c > 0$  and  $k(n) \geq O(\log n)^{c+3}$ , we can have any  $m'(n) = m(n) - r(n) \leq \alpha \cdot k(n)$  with  $r(n) \cdot k(n) = m'(n) \cdot n / (\log n)^c$ . In particular:*
  - *We can have  $m'(n) = \alpha \cdot k(n)$  and  $r(n) = n / (\log n)^c$ .*
  - *We can have  $m'(n) = \min(\alpha \cdot k(n), (\log n)^c)$  and  $r(n) = n/k(n)$ .**In addition, for any  $k(n) = \Omega(\log n)$  we can have  $m'(n) = m(n) - r(n) = \min(\alpha \cdot k(n), \text{poly}(\log n))$  and  $r(n) = n$ .*

Note that in Part 1 the output length exceeds the seed length, whereas in the other parts the output is shorter. The extractors in Parts 1 and 2 are strong, but this is not the case (and cannot be the case) in Part 3. The additional claim (in Part 3) is merely a restating of Theorem 6.1, which also appears as a special case of Corollary 6.2.

**Proof.** In all parts,  $t = t(n)$  denotes the number of times that the basic extractor  $E_0 : \{0, 1\}^n \times \{0, 1\}^{r_0(n)} \rightarrow \{0, 1\}^{m_0(n)}$  is invoked; that is, we will derive  $m(n) = t(n) \cdot m_0(n)$  and  $r(n) = t(n) \cdot r_0(n)$ .

Part 1 (resp., Part 2) is obtained by combining Theorem 2.4 (resp., Theorem 3.1) with the second part of Theorem 6.3 (i.e., the part referring to strong extractors). In both parts we have  $r_0(n) = \Theta(\log n)$ ; in Part 1 we can have any  $m_0(n) = r_0/\beta = O(r_0(n))$ , whereas in Part 2 we have some  $m_0(n) = \Omega(r_0(n))$ . Note that the extractor asserted in Theorem 2.4 can be implemented by depth-three  $\mathcal{AC}^0$  circuits.

Part 3 combines Corollary 6.2 with the first part of Theorem 6.3 (i.e., the part referring to ordinary extractors). Here we use  $r_0 = \min(n, O(n \log^3 n)/k(n))$  and  $m'_0 = m_0 - r_0 = \min(\alpha k(n), (\log n)^{c+3})$ . Hence, for  $k(n) \geq O(\log n)^{c+3}$ , we have  $r_0 \cdot k(n) = m'_0 \cdot n / O(\log n)^c$ , which yields  $r(n) \cdot k(n) = m'(n) \cdot n / O(\log n)^c$  for  $r(n) = t(n) \cdot r_0(n)$  and  $m'(n) = t(n) \cdot m'_0(n) \leq \alpha k(n)$ . ◀

**7 Extraction from several independent sources**

While deterministic extraction is not possible from a single general  $(n, n-1)$ -source, it is known to be possible when having a constant number of independent sources (cf., e.g., [17, 7, 8, 38]). We ask whether such extraction is possible in  $\mathcal{AC}^0$ , provided that the min-entropy rate is at least  $1/\text{poly}(\log n)$ . Considering extraction from a constant number of independent sources, note that the impossibility results in Section 5.2 can be adapted to rule out min-entropy rates below  $1/\text{poly}(\log n)$ . Details follow.

We view the  $c$  sources as a single source of length  $cn$  in which the  $n$ -bit long parts are independent of one another. Applying the proof of Theorem 5.4 to such a generic source,

yields a bit-fixing source of length  $cn$  in which  $ck$  bits are random. These  $ck$  positions are “typical” (w.r.t an averaging argument) and so we can make sure that there are approximately  $k$  random bits in each  $n$ -bit long part of the source. This proves that  $\mathcal{AC}^0$ -extraction from a constant number of sources is impossible if all sources have min-entropy rate below  $n/\text{poly}(\log n)$ . However, above that level of min-entropy,  $\mathcal{AC}^0$ -extraction is possible when *using a seed of logarithmic length*. Hence, it is reasonable to ask whether deterministic  $\mathcal{AC}^0$ -extractors can meet this performance (as in the case of bit-fixing sources).

## 7.1 Extraction from two independent sources

In this section, we consider  $\mathcal{AC}^0$ -extractors for pairs of independent sources, a model first considered by Vazirani [56] (for Santha-Vazirani sources [52]). Recall that Chor and Goldreich [17] showed that inner-product mod 2 yields a good extractor for pairs of independent sources provided that each source has min-entropy rate above half; in fact, it yields an  $\epsilon$ -extractor for any pair consisting of an  $(n, k_1)$ -source and an  $(n, k_2)$ -source such that  $k_1 + k_2 > n + 1 + 2 \log(1/\epsilon)$ . A natural question that arises is whether such a seedless extractor can operate in  $\mathcal{AC}^0$ . Specifically:

► **Definition 7.1** (a (seedless) two-source extractor). The function  $E : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}^m$  is called a  $((k_1, k_2), \epsilon)$ -extractor if for every pair of independent random variables  $(X, Y)$  such that  $X$  is a  $(n, k_1)$ -source and  $Y$  is a  $(n, k_2)$ -source it holds that  $\Delta[E(X, Y); U_m] \leq \epsilon$ .

We ask whether a  $((0.51n, 0.51n), 0.01)$ -extractor can be computed in  $\mathcal{AC}^0$ . So far, we only know of  $((k, k), 0.01)$ -extractors in  $\mathcal{AC}^0$  for  $k(n) = n - (n/\log^4 n)$  and  $m = 1$ .

► **Theorem 7.2** (two-source deterministic extractor in  $\mathcal{AC}^0$ ). For  $k(n) = n - (n/\log^4 n)$  and  $\epsilon(n) = n^{-\omega(1)}$ , there exists a  $((k, k), \epsilon)$ -extractor in  $\mathcal{AC}^0$  (extracting a single bit).

The constant 4 in the exponent is an arbitrary constant greater than three.

**Proof.** First, we shall see how to extract a single bit with error  $O(1/\log n)$  from sources of rate  $1 - \log^{-3} n$ . Consider a partition of each of the two sources into blocks of length  $\ell = \log^2 n$ ; that is, let  $X = (X_1, \dots, X_{n/\ell})$  and  $Y = (Y_1, \dots, Y_{n/\ell})$  such that  $|X_i| = |Y_i| = \ell$ . We shall prove (see Claim 7.3 below) that each source is  $\exp(-\Omega(\ell))$ -close to a source in which at least a  $1 - 10 \log^{-3} n$  fraction of the blocks have min-entropy rate at least 0.7 conditioned on the previous blocks.

Next, we apply the inner-product mod 2 extractor [17] on each pair of corresponding blocks, obtaining a sequence  $Z$  of  $n' \stackrel{\text{def}}{=} n/\ell$  bits such that at least  $1 - 20 \log^{-3} n$  of them are “quasi-random” conditioned on the previous bits; that is,  $Z_i$  is the inner-product mod 2 of  $X_i$  and  $Y_i$  and for all but at most  $20n'/\log^3 n$  of the indices  $i \in [n']$  it holds that  $\Pr[Z_i = 1 | Z_{[i-1]} = \alpha] = 0.5 \pm \exp(-\Omega(\ell))$  for every  $\alpha \in \{0, 1\}^{i-1}$ . Applying the extractor of Theorem 5.7 (which was used in the proof of Theorem 5.8) to  $Z$ , we extract a bit with bias  $O(1/\log n)$ . (Indeed, a sequence as above is  $n \cdot \exp(-\Omega(\ell))$ -close to a *non-oblivious* bit-fixing source of length  $n/\ell$  with  $20n/(\ell \log^3 n)$  bits that are fixed as an arbitrary function of the random bits.)<sup>33</sup>

Finally, to reduce the error of the extractor, we use a source of higher min-entropy rate  $(1 - \log^{-4} n)$ , which we break into  $\log n$  parts such that each part has conditional min-entropy

<sup>33</sup> Given a sequence of  $n' = n/\ell$  bits in which  $k'$  bits are quasi-random conditioned on the previous bits, note that the subsequence of quasi-random bits is close to being uniformly distributed in  $\{0, 1\}^{k'}$ .



rate at least  $1 - \log^{-3} n$ . Applying the foregoing extractor to each part, we get  $\log n$  bits such that each bit has bias  $O(1/\log n)$  conditioned on the previous ones. XORing these bits, we obtain a bit with bias  $O(1/\log n)^{\log n} = n^{-\omega(1)}$ . The proof is completed once we prove the following claim.

► **Claim 7.3** (on the number of blocks with high conditional min-entropy). *Let  $X = (X_1, \dots, X_{n'})$  be a  $(n, \delta n)$ -source such that  $|X_i| = \ell = n/n'$ , and  $\delta', \epsilon' \in (0, 1)$  such that  $\epsilon'\ell \geq \log n$ . Then,  $X$  is  $\exp(-\Omega(\epsilon'\ell))$ -close to a source  $X'$  for which there exists a set  $I \subseteq [n']$  of cardinality at least  $\frac{\delta - \delta'}{1 - \delta'} \cdot n'$  such that for every  $i \in I$  and  $x \in \{0, 1\}^n$  it holds that  $\Pr[X'_i = x_i | X'_{[i-1]} = x_{[i-1]}] \leq 2^{-(\delta' - 2\epsilon')\ell}$ .*

For the above application we set  $\delta = 1 - \log^{-3} n$ ,  $\delta' = 0.9$ , and  $\epsilon' = 0.1$ . Hence,  $\frac{\delta - \delta'}{1 - \delta'} = 1 - \frac{1 - \delta}{1 - \delta'} = 1 - 10 \cdot (1 - \delta)$  and  $\delta' - 2\epsilon' = 0.7$ .

**Proof.** As in many known cases, the proof proceeds via analyzing the collision probability. Recall that the collision probability of a random variable  $Z$ , denoted  $\mathbf{CP}[Z]$ , equals  $\Pr[Z^{(1)} = Z^{(2)}]$ , where  $Z^{(1)}$  and  $Z^{(2)}$  are two independent copies of the random variable  $Z$ ; that is,  $\mathbf{CP}[Z] = \sum_z \Pr[Z = z]^2$ . Using the fact that  $X$  has min-entropy  $\delta n$ , we infer that  $\mathbf{CP}[X] \leq 2^{-\delta n}$ . Hence, it holds that

$$\prod_{i \in [n']} \Pr[X_i^{(1)} = X_i^{(2)} | X_{[i-1]}^{(1)} = X_{[i-1]}^{(2)}] \leq 2^{-\delta n' \cdot \ell}, \tag{24}$$

where  $X^{(1)}$  and  $X^{(2)}$  are two independent copies of  $X$ . We call  $i$  good if  $\Pr[X_i^{(1)} = X_i^{(2)} | X_{[i-1]}^{(1)} = X_{[i-1]}^{(2)}] \leq 2^{-\delta'\ell}$ , and infer that at least a  $\frac{\delta - \delta'}{1 - \delta'}$  fraction of the  $i$ 's are good.<sup>34</sup> We next observe that, for every good  $i$ , there exists a set  $S_{i-1} \subseteq \{0, 1\}^{(i-1)\ell}$  such that  $\Pr[X_{[i-1]} \in S_{i-1}] \geq 1 - 2^{-\epsilon'\ell}$  and for every  $z \in S_{i-1}$  it holds that

$$\Pr[X_i^{(1)} = X_i^{(2)} | X_{[i-1]}^{(1)} = X_{[i-1]}^{(2)} = z] \leq 2^{-(\delta' - \epsilon')\ell}. \tag{25}$$

Moving back to a min-entropy upper bound, for any good  $i$  and every  $z \in S_{i-1}$ , let  $H_{i,z} \stackrel{\text{def}}{=} \{x_i : \Pr[X_i = x_i | X_{[i-1]} = z] > 2^{-(\delta' - 2\epsilon')\ell}\}$ . Then,  $\Pr[X_i \in H_{i,z} | X_{[i-1]} = z] \leq 2^{-\epsilon'\ell}$  holds (for any good  $i$  and  $z \in S_{i-1}$ ).<sup>35</sup> Let  $X' = X$  if for every good  $i$  it holds that  $X_{[i-1]} \in S_{i-1}$  and  $X_i \in H_{i,X_{[i-1]}}$ , and  $X'$  be uniform otherwise. Then,  $X'$  is  $O(n' \cdot 2^{-\epsilon'\ell})$ -close to  $X$ , and for every good  $i$  and every  $x \in \{0, 1\}^n$  it holds that  $\Pr[X'_i = x_i | X'_{[i-1]} = x_{[i-1]}] \leq 2^{-(\delta' - 2\epsilon')\ell}$ . The claim follows. ◀

This completes the proof of the theorem. ◀

<sup>34</sup> Denoting the set of good  $i$ 's by  $G$ , we have  $(2^{-\ell})^{|G|} \cdot (2^{-\delta'\ell})^{n' - |G|} \leq 2^{-\delta n' \cdot \ell}$ , and it follows that  $(1 - \delta')^{|G|} \geq (\delta - \delta')n'$ .

<sup>35</sup> Let  $Y$  and  $Y'$  be distributed independently and identically to  $X_i$  conditioned on  $X_{[i-1]} = z$  (i.e.,  $\Pr[Y = y] = \Pr[X_i = y | X_{[i-1]} = z]$ ), and  $H \stackrel{\text{def}}{=} \{y : \Pr[Y = y] > 2^{-(\delta' - 2\epsilon')\ell}\}$ . Then,  $\Pr[Y = Y'] \leq 2^{-(\delta' - \epsilon')\ell}$  implies  $\Pr[Y \in H] \leq 2^{-\epsilon'\ell}$ . See farther discussion in Remark 7.1.

### Digest

Note that the sequence  $Z$  extracted at the first part (of the main step) is apparently more restricted than a non-oblivious bit-fixing source, but it is not an (oblivious) bit-fixing source. Hence, we applied the extractor of Theorem 5.7 to  $Z$ , rather than applying the  $\mathcal{AC}^0$ -extractors for bit-fixing sources that can handle lower min-entropy. Of course, there is room in between, and one may capitalize on it, but currently we do not see a way of doing so. Indeed, the point is trying to obtain  $\mathcal{AC}^0$ -extraction for two independent sources of some constant min-entropy rate. (In Section 7.2, we achieve the corresponding goal for four sources.)

► Remark (from conditional collision probability to conditional min-entropy). The proof of Claim 7.3 relies on transforming an upper bound on conditional collision probability to a lower bound on conditional min-entropy. For the benefit of future applications, we distill the corresponding claim here.

*Let  $X^{(1)}$  and  $X^{(2)}$  be two independent copies of  $X$ , and let  $f, g : \{0, 1\}^* \rightarrow \{0, 1\}^*$ . If  $\Pr[f(X^{(1)}) = f(X^{(2)}) | g(X^{(1)}) = g(X^{(2)})] \leq \epsilon \cdot 2^{-k}$ , then  $X$  is  $2\sqrt{\epsilon}$ -close to  $X'$  such that  $\Pr[f(X') = v | g(X') = u] \leq 2^{-k}$  for every  $u, v$ .*

The proof of this claim proceeds in two steps. First, we infer that there exists a set  $S$  such that  $\Pr[g(X) \in S] \geq 1 - \sqrt{\epsilon}$  and  $\Pr[f(X^{(1)}) = f(X^{(2)}) | g(X^{(1)}) = g(X^{(2)}) = s] \leq \sqrt{\epsilon} \cdot 2^{-k}$  for every  $s \in S$ . Next, for every  $s \in S$  and every  $v$ , we show that  $X$  is  $\sqrt{\epsilon}$ -close to  $X'$  that satisfies  $\Pr[f(X') = v | g(X') = s] \leq 2^{-k}$ . This is shown by considering the random variable  $X_s$  defined as  $f(X)$  conditioned on  $g(X) = s$ , letting  $H = \{v : \Pr[X_s = v] > 2^{-k}\}$ , and noting that  $\mathbf{CP}[X_s] \geq \Pr[X_s \in H] \cdot \min_{v \in H} \{\Pr[X_s = v]\}$ , which implies  $\sqrt{\epsilon} 2^{-k} \geq \mathbf{CP}[X_s] > \Pr[X_s \in H] \cdot 2^{-k}$ . The claim follows by letting  $X'$  be  $X$  conditioned on both  $g(x) \in S$  and  $f(X) \notin H$ .

► Remark (reducing two-source extraction to extraction from non-oblivious block-fixing sources). As stated above, the core of the proof of Theorem 7.2 is a transformation of a pair of independent sources, each having min-entropy rate  $\delta > 0.75$ , into a non-oblivious bit-fixing source of min-entropy rate at least  $1 - 4 \cdot (1 - \delta) - o(1)$ , where here we use Claim 7.3 with any constant  $\delta' > 0.5$  and  $\epsilon' = 1/\log n$  (rather than with  $\delta' = 0.9$  and  $\epsilon' = 0.1$ ). Similarly, we can transform such a pair of sources into a non-oblivious block-fixing source with block length  $\Omega(\log^2 n)$ , by using an extractor that outputs  $\Omega(\ell)$  bits [17, 20]. Hence, deterministic two-source  $\mathcal{AC}^0$ -extractors for some constant min-entropy rate exist if such extractors exist for a non-oblivious block-fixing source of some constant rate and super-logarithmic block length.

Recall that there exist no deterministic extractors for non-oblivious *bit*-fixing source of any (non-trivial) constant min-entropy rate [40]. Such a result is not known for extraction from non-oblivious *block*-fixing source of (any constant min-entropy rate and) super-logarithmic block length, although it was conjectured more than a decade ago in [23, Conj. 2.2]. Hence, ruling out two-source extractors for any constant rates requires settling the latter conjecture.

## 7.2 Extraction from four independent sources

In this section, we consider  $\mathcal{AC}^0$ -extractors from a constant number of independent sources, showing that such extractor exist for *four* independent sources of *some* constant min-entropy rate. In Section 7.3, we shall show how to extract from independent sources of any constant min-entropy rate, but we shall use a larger number of sources (which depends on the rate). In both section, we shall refer to the following definition, which generalizes Definition 7.1.

► **Definition 7.4** (a (seedless) multi-source extractor). For a constant integer  $c \geq 2$ , the function  $E : (\{0, 1\}^n)^c \rightarrow \{0, 1\}^m$  is called a  $((k_1, \dots, k_c), \epsilon)$ -extractor if for every sequence of  $c$  independent random variables  $(X^{(1)}, \dots, X^{(c)})$  such that  $X^{(i)}$  is a  $(n, k_i)$ -source for  $i = 1, \dots, c$ , it holds that  $\Delta[E(X^{(1)}, \dots, X^{(c)}); U_m] \leq \epsilon$ .

Indeed, Definition 7.1 corresponds to the special case of  $c = 2$ .

While Theorem 7.2 applies to (two) sources of min-entropy rate of the form  $1 - o(1)$ , the following result holds for (four) sources of min-entropy rate that is bounded away from 1. Furthermore, while Theorem 7.2 does not assert *uniform*  $\mathcal{AC}^0$  circuits, the following result does assert such circuits.

► **Theorem 7.5** (four-source deterministic extractor in  $\mathcal{AC}^0$ ). For  $k(n) = 0.99n$  and  $\epsilon(n) = n^{-\omega(1)}$ , there exists a  $((k, k, k, k), \epsilon)$ -extractor in *uniform*  $\mathcal{AC}^0$  (extracting poly-logarithmically many bits).

**Proof Sketch.** The general strategy is to use the first two sources in order to sample polylogarithmically many bits of the third and fourth sources, and then extract out of the selected sub-sources. Thus, in a sense, we construct a condenser that transforms the second pair of sources into a pair of short sources that maintains a sufficiently high min-entropy rate; specifically, the resulting sources have min-entropy rate at least 0.51 (whereas the original ones had min-entropy rate 0.99). This conversion is performed using the imperfect randomness that exists in the first pair of sources.

Specifically, we will consider a partition of each source into  $n' = n/\ell$  blocks of length  $\ell = \log^2 n$ , denoted  $X^{(i)} = (X_1^{(i)}, \dots, X_{n'}^{(i)})$ , and apply a two-source extractor  $E' : \{0, 1\}^\ell \times \{0, 1\}^\ell \rightarrow [n'/\ell]$ , which is computable by constant-depth  $\text{poly}(n)$ -size circuits (e.g.,  $E'$  computes bilinear functions [17, 20]), to the blocks of the first two sources. We view these  $n'$  outcomes (i.e.,  $E'(X_j^{(1)}, X_j^{(2)})$  for  $j \in [n']$ ) as an assignment of blocks indexed by  $[n']$  into  $n'' = n'/\ell$  cells (i.e., block  $j$  is assigned to cell  $E'(X_j^{(1)}, X_j^{(2)})$ ), and pick a cell  $c_{\min} \in [n'/\ell]$  with a minimal number of blocks (breaking ties arbitrarily).<sup>36</sup>

Finally, we apply a second two-source extractor  $E'' : \{0, 1\}^{\ell^2} \times \{0, 1\}^{\ell^2} \rightarrow \{0, 1\}^{\Omega(\ell)}$  to the sub-sources  $X_{j_1}^{(3)} \dots X_{j_{\ell'}}^{(3)}$  and  $X_{j_1}^{(4)} \dots X_{j_{\ell'}}^{(4)}$ , where  $(j_1, \dots, j_{\ell'})$  is the sequence of blocks assigned to cell  $c_{\min}$  and  $\ell' \leq \ell$ . (If  $\ell' < \ell$ , then we pad the said sequence of blocks to the full length of  $\ell^2$ .)

Intuitively, the first extractor  $E'$  uses blocks in the first pair of sources (i.e.,  $X^{(1)}$  and  $X^{(2)}$ ) in order to assign blocks of the second pair of sources into cells. Pairs of blocks of sufficiently high min-entropy (in  $X^{(1)}$  and  $X^{(2)}$ ) will assign the corresponding blocks (of  $X^{(3)}$  and  $X^{(4)}$ ) to a random cell, but other pairs of blocks (in  $X^{(1)}$  and  $X^{(2)}$ ) may assign blocks (of  $X^{(3)}$  and  $X^{(4)}$ ) arbitrarily. Still, each cell is assigned many blocks that have sufficiently high min-entropy in the second pair of sources. Hence, the smallest cell, denoted  $c_{\min}$ , contains blocks of  $X^{(3)}$  and  $X^{(4)}$  that have average min-entropy rate that exceeds 0.5, and applying the extractor  $E''$  to these blocks (of  $X^{(3)}$  and  $X^{(4)}$ ) will yield an almost random output. Note that the value  $c_{\min}$  may not be random; it is merely determined as the index of the smallest cell.

<sup>36</sup> Indeed, this strategy is inspired by Feige’s protocol for leader election [22], and it was already employed in the context of  $O(1)$ -source extraction by Li [38]. Here, we pick the smallest cell in order to guarantee that the total length of the blocks assigned to it is small. This guarantees both that  $E''$  can be computed by constant-depth circuits of  $\text{poly}(n)$ -size and that the min-entropy rate of blocks (of  $X^{(3)}$  and  $X^{(4)}$ ) in this cell exceeds half. Jumping ahead, we mention that in the proof of Theorem 7.6 we shall only rely on the first implication (and the second implication will not hold).

In order to analyze the quality of this construction, we first invoke Claim 7.3. Specifically, for  $\delta = 0.99$  and some constant  $\delta', \epsilon' \in (0, 1)$ , we say that a block index  $j \in [n']$  is **good** for the  $i^{\text{th}}$  source if  $\Pr[X'_j = x_j | X'_{[j-1]} = x_{[j-1]}] \leq 2^{-(\delta' - 2\epsilon')\ell}$ , for every  $x$ , where  $X'$  is  $\exp(-\Omega(\epsilon'\ell))$ -close to  $X^{(i)}$ . Recall that at least a  $\frac{\delta - \delta'}{1 - \delta'} = 1 - \frac{1 - \delta}{1 - \delta'}$  fraction of the blocks are good for each source. We call  $j$  **good** if it is good for all four sources, and conclude that at least a  $1 - 4 \cdot \frac{1 - \delta}{1 - \delta'} > 1/2$  fraction of the blocks are good. Hence, with very high probability, each cell is assigned at least  $(1 - 4 \cdot \frac{1 - \delta}{1 - \delta'} - o(1)) \cdot \ell$  good blocks, and the min-entropy of the blocks assigned to each cell is at least  $(1 - 4 \cdot \frac{1 - \delta}{1 - \delta'} - o(1)) \cdot \ell \cdot (\delta' - 2\epsilon')\ell$ . The analysis is completed by selecting  $\delta'$  and  $\epsilon'$  such that  $(1 - 4 \cdot \frac{1 - \delta}{1 - \delta'}) \cdot (\delta' - 2\epsilon')$  is strictly larger than  $1/2$  (e.g., using  $\delta' = 0.9$  and  $\epsilon' = 0.01$  we get a lower bound of 0.52).

We wish to emphasize a key point regarding the foregoing probabilistic analysis. Recall that blocks are defined as “good” (for a particular source) based on their *conditional* min-entropy. That is, the  $j^{\text{th}}$  block (of the  $i^{\text{th}}$  source) is good if the conditional min-entropy of that block, conditioned on the prior  $j - 1$  blocks of this source, exceeds a specific threshold. It follows that if the  $j^{\text{th}}$  block is good for both the first and second sources, then the  $j^{\text{th}}$  block (of the third and fourth sources) will be assigned a random cell, conditioned on the assignment of the prior  $j - 1$  blocks. Similarly, the blocks assigned to the smallest cell (i.e.,  $c_{\min}$ ) are defined as good in the same sense, which implies that their total min-entropy is sufficiently high, since it is the sum of their individual conditional min-entropies. Hence, if a  $\rho$  fraction of the blocks assigned to the smallest cell are good, then each of these blocks has conditional min-entropy rate at least  $\delta'' = \delta' - 2\epsilon'$  (conditioned on the prior blocks), and the total min-entropy rate of the blocks in this cell is at least  $\rho \cdot \delta''$ .

What remains is implementing the determination the smallest cell and its contents by constant-depth circuits of  $\text{poly}(n)$ -size. This can be done using techniques as in the proof of Theorem 3.8. Specifically, we refer to the ranking procedure applied in Step 3(b) of the proof.<sup>37</sup> Lastly, note that the foregoing extractor outputs  $\Omega(\ell)$  bits, where  $\ell = \log^2 n$ . The proof remains intact when setting  $\ell$  to be any larger poly-logarithmic function. ◀

### 7.3 Extraction from $\text{poly}(1/\delta)$ sources of rate $\delta$

Looking at the four-source extractor (presented in the proof of Theorem 7.5), one may notice that there are two main reasons for the high constant lower bound on the min-entropy rate (i.e., 0.99) used there. One is that we used a bilinear two-source extractor, whereas we have such extractors only for rate greater than 0.5 (and, in general, explicit two-source extractors are known [13] only for constant rate that is slightly smaller than 0.5).<sup>38</sup> The second reason is that we use blocks that are “good” (i.e., have sufficient min-entropy rate) with respect to *all* sources.

But if we are willing to use more sources, we may reach an arbitrary low constant rate. Firstly, we will use the multi-source extractors of Barak *et al.* [7], which can extract from any constant rate  $\rho > 0$  using  $\text{poly}(1/\rho)$ -many sources of such rate. The crucial observation is that these extractors compute polynomials (of degree that is smaller than the number of sources) over a finite field, and they can be computed by constant-depth circuits of sub-exponential size. Since we shall be applying these extractors to sub-sources of polylogarithmic (in  $n$ ) length, we can afford this size, which is  $\text{poly}(n)$ .

<sup>37</sup> Recall that our ranking procedure amount to counting (in uniform  $\mathcal{AC}^0$ ) the number of marked elements in an array of length  $n'$ , when guaranteed that this number does not exceed  $\text{poly}(\log n)$ . An explicit construction of such a counter was presented in [48], improving over [3, Sec. 5] (and subsequent works).

<sup>38</sup> The said constant is not specified in [13], and is estimated to be higher than 0.49.

Secondly, there is no need to insist on using only blocks that are good for all sources; we can use blocks that are good for a constant fraction of the sources. This assertion relies on the fact that the extractors of [7] work well also when applied to many (independent) sources such that only a constant fraction of these sources are good (i.e., have sufficient min-entropy). The latter fact is based on two observations:

1. The extractors of [7] iterate the three-source extractor  $E_3(x, y, z) = xy + z$ , where  $x, y, z$  are field elements. Hence, assuming that  $\Pr[X=0] = 0$  and ditto for  $Y$ , the min-entropy of  $E_3(X, Y, Z)$  is at least the maximum among the min-entropy of the three sources. (The condition can be guaranteed by redefining each of the sources so that it never assumes the value zero, while noting that this reduces the min-entropy of the source by at most one unit.)
2. By the entropy increasing lemma of [7], the min-entropy of  $E_3(X, Y, Z)$  is at least a constant factor larger than the minimum among the min-entropy of the three sources, as long as this value does not exceed  $0.9\ell$ , where  $2^\ell$  is the size of the field. Using the following tree marking game, this implies that the iterative extraction procedure of [7], which applies  $E_3$  at the internal nodes of a ternary tree (while the sources are placed at the leaves), produces output of min-entropy at least  $0.9\ell$ .

Consider a full ternary tree of height  $h$  such that a  $\rho$  fraction of the leaves are marked 1 (corresponding to good sources of sufficiently high min-entropy rate), and the other leaves are marked 0 (corresponding to bad sources). Going from the leaves to the root, the following marking rule is applied: If the children of a node are all marked  $i > 0$ , then the parent is marked  $i + 1$  (corresponding to an application of the entropy increasing lemma of [7]), otherwise the parent is marked by the maximum of its children (corresponding to preservation of entropy asserted in Observation 1). Then, as shown in Claim 7.7 (below), for every fixed  $\rho > 0$  and any (allowed) marking of the leaves, the minimal possible marking of the root is unbounded as a function of  $h$ .

Plugging these observations into the framework of the proof of Theorem 7.5, while replacing each pair of sources by many independent sources, we get uniform  $\mathcal{AC}^0$ -extractors for  $\text{poly}(1/\delta)$ -sources of min-entropy rate  $\delta$ , for any  $\delta > 0$ .

► **Theorem 7.6** (deterministic extractor in  $\mathcal{AC}^0$  for  $\text{poly}(1/\delta)$  sources of constant rate  $\delta$ ). *For any constants  $\delta > 0$  and  $\gamma > 1$ , setting  $t = \text{poly}(1/\delta)$ , there exists a uniform  $\mathcal{AC}^0$  function  $E : \{0, 1\}^n \rightarrow \{0, 1\}^{\Theta(\log^\gamma n)}$  that constitutes a  $((\delta n, \dots, \delta n), n^{-\omega(1)})$ -extractor.*

**Proof Sketch.** We partition the  $t$  sources into two sets; a set of  $t_1$  sources will be used to select a small cell, and the remaining  $t_2 = t - t_1$  sources will be used for the actual extraction based on sub-sources determined by the selected cell. That is, as in the proof of Theorem 7.5, we partition each source into  $n' = n/\ell$  blocks, each of length  $\ell = \log^\gamma n$ , and use two multi-source extractors, denoted  $E'$  and  $E''$ , for sources of length  $\ell$  and  $\ell^2$ , respectively. However, here the first extractor (i.e.,  $E'$ ) uses  $t_1 \gg 2$  sources and the second extractor (i.e.,  $E''$ ) uses  $t_2 \gg 2$  sources. Furthermore, here  $E'$  extracts a slightly longer string, which is interpreted as an element in  $[n'] \times [n'/\ell]$ . Specifically, such  $(b, c) \in [n'] \times [n'/\ell]$  is interpreted as placing the  $b^{\text{th}}$  block in cell number  $c$ . That is, the pair extracted from the  $j^{\text{th}}$  block (of the first  $t_1$  sources) does not assign block  $j$  to some cell, but rather assigns a hopefully random block to a hopefully random cell, where these hopes are materialized if extraction from the  $j^{\text{th}}$  block succeeds (i.e., if this block has sufficient min-entropy in sufficiently many sources).

Specifically, the first stage of the extractor will use the iterated extraction procedure of [7]. Recall that this procedure is based on the extractor  $E_3 : F^3 \rightarrow F$  that operates over the finite

field  $F$  (of prime cardinality) such that  $E_3(x, y, z) = xy + z$  and  $|F| \approx 2^\ell$ . Actually, we shall use a minor modification of  $E_3$  that avoids zero elements by letting  $E'_3(x, y, z) = g(x)g(y) + g(z)$ , where  $g(0) = 1$  and  $g(x) = x$  for all  $x \in F \setminus \{0\}$ . Note that the behaviour of  $E'_3$  with respect to sources of min-entropy  $k$  is captured by the behaviour of  $E_3$  with respect to sources of min-entropy  $k - 1$ , since  $\mathbf{H}_\infty(g(X)) \geq \mathbf{H}_\infty(X) - 1$ , where  $\mathbf{H}_\infty(V)$  denotes the min-entropy of the random variable  $V$ . The extractor  $E' : F^{3^i} \rightarrow F$  is defined recursively (as in [7]) by  $E'(x_1, \dots, x_{3^i}) = E'_3(y_1, y_2, y_3)$ , where  $y_j = E'(x_{(j-1) \cdot 3^{i-1} + 1}, \dots, x_{j \cdot 3^{i-1}})$ . Actually,  $E'(x_1, \dots, x_{t_1})$  is redefined as the first  $2 \log n' - \log \ell$  bits of  $E'(x_1, \dots, x_{t_1})$  (as defined above). Recall that  $E'$  will be applied  $n'$  times, where the  $j^{\text{th}}$  application takes as input  $(X_j^{(1)}, \dots, X_j^{(t_1)})$ , where  $X_j^{(i)}$  is the  $j^{\text{th}}$  block of the  $i^{\text{th}}$  source. We shall detail the second stage of the final extractor at a later point.

Towards analyzing this construction, we first invoke Claim 7.3 (using  $\delta' = \delta/2$  and  $\epsilon' = \delta/12$ ). Saying that a block  $j$  is good for the  $i^{\text{th}}$  source if its conditional min-entropy rate exceeds  $\delta/3$ , we infer that at least  $\delta/2$  of the blocks are good for each specific source (i.e., the  $i^{\text{th}}$  source). (Actually, the bound refers to the conditional min-entropy of a source that is close to the  $i^{\text{th}}$  source.) Hence, at least  $\delta/4$  of the blocks are good for at least  $\delta/4$  of the (first  $t_1$ ) sources. Let us call such a block good, and analyze extraction from it using the two foregoing observations. We establish the second observation by proving the following.

► **Claim 7.7** (analysis of the tree marking game). *Consider a full ternary tree of height  $h$  such that at least  $\rho \cdot 3^h$  leaves are assigned the value 1 and the other leaves are assigned the value 0. Assign an internal node the value  $i + 1$  if all its children are assigned the value  $i > 0$ , and otherwise assign it the maximum value that is assigned to its children. Then, for every fixed  $\rho \geq 0.9^h$ , the value assigned to the root is at least  $h/100$ .*

**Proof.** Let us turn the table around and denote by  $N_h(i)$  the maximal number of leaves that may be assigned the value 1 in a ternary tree in which the root is assigned a value that is smaller or equal to  $i$ . Clearly,  $N_h(i) \geq N_h(i - 1)$  and  $N_{h+1}(i) \geq N_h(i)$ . Note that for a tree consisting of a single vertex (i.e.,  $h = 0$ ), it holds that  $N_0(0) = 0$  and  $N_0(i) = 1$  for every  $i \geq 1$ . The key observation is that

$$N_{h+1}(i) = \max(3 \cdot N_h(i - 1), 2 \cdot N_h(i) + N_h(i - 1)) = 2 \cdot N_h(i) + N_h(i - 1).$$

It follows that  $N_h(i) = 2^h + \sum_{j \in [h]} 2^{j-1} \cdot N_{h-j}(i - 1)$ , for every  $i \geq 1$ . Next, for  $h, i \geq 1$ , one can prove by induction on  $i$  that  $N_h(i) = \sum_{j=0}^{i-1} 2^{h-j} \cdot \binom{h}{j}$ . Our last technical claim is that  $N_h(i) \geq 2.7^h$  implies  $i \geq h/100$ . Suppose, towards the contradiction that  $i < h/100$ . Then,  $N_h(i) = \sum_{j=0}^{i-1} 2^{h-j} \cdot \binom{h}{j} < 2^h \cdot 2^{H_2(0.01) \cdot h}$ , which is upper bounded by  $2^{h+0.1h} < 2.2^h$ , in contradiction to  $N_h(i) \geq 2.7^h$ . Turning back to the main claim, we note that if a  $\delta \geq 0.9^h$  fraction of the leaves are assigned the value 1 and the root is assigned the value  $i$ , then  $N_h(i) \geq \delta \cdot 3^h$ , which implies that  $i \geq h/100$ . ◀

Now, assuming that the  $j^{\text{th}}$  block is good, we claim that  $E'(X_j^{(1)}, \dots, X_j^{(t_1)})$  is  $\exp(-\Omega(\ell))$ -close to the uniform distribution over  $[n'] \times [n'/\ell]$  (independently of prior blocks). This is the case since the  $j^{\text{th}}$  block is good for at least an  $\delta/4$  fraction of the sources, whereas the values analyzed in Claim 7.7 (with  $\rho = \delta/4$ ) reflect the min-entropy of the corresponding extracted field element. Specifically, leaves assigned the value 1 correspond to blocks with (conditional) min-entropy rate of at least  $\delta/3$ ; internal nodes with value  $i + 1$  having children that have value  $i > 0$  correspond to the application of  $E'_3$  to random variables of min-entropy rate at least  $\min((1 + \Omega(1))^i \cdot \delta/3, 0.9)$  (see [7, Lem. 3.1]); whereas the other internal nodes correspond to the application of  $E'_3$  that maintain the maximum min-entropy rate of the



three random variables to which  $E'_3$  is applied.<sup>39</sup> Hence, using  $h = \Theta(\log(1/\delta))$ , we obtain at level  $h - 1$  of the tree (i.e., at the children of the root) three independent sources such that each of them has (conditional) min-entropy rate at least 0.9, where the conditioning is over the values of prior blocks. Using [43, Lem. 13], while relying on the fact that  $E_3(r, y, s)$  may be viewed as a universal hashing function mapping  $y$  to  $ry + s$  (based on the key  $(r, s)$ ), we are done.

Again, we wish to emphasize a key point regarding the foregoing probabilistic analysis. Recall that blocks are defined as good for a specific source based on their *conditional* min-entropy in that source. A block is globally good if it is good for sufficiently many sources (out of the first  $t_1$  sources). Hence, extraction from a good block yields a quasi-random outcome, conditioned on the values of all prior blocks (and the corresponding outcomes that were extracted). As detailed next, a similar analysis will apply to the blocks (of the remaining  $t_2$  sources) assigned to any specific cell.

It is time to detail the second stage of the final extractor. This stage uses the remaining  $t_2 = t - t_1$  sources. First, the  $n'$  outcomes obtained in the first stage are interpreted as an assignment of blocks to cells; specifically, block  $b$  is assigned to cell  $c$  if the pair  $(b, c) \in [n'] \times [n'/\ell]$  appears in this  $n'$ -long sequence of outcomes. Then, a cell with a minimal number of assigned blocks is picked, and the extractor  $E''$  (which is analogous to the extractor  $E'$  that was used in the first stage) is applied to the corresponding  $t_2$  sub-sources; specifically, if the selected cell was assigned the blocks  $j_1, \dots, j_{\ell'}$ , where  $\ell' \leq \ell$ , then  $E''$  is applied to the  $t_2$  sub-sources  $X_{j_1}^{(t_1+1)} \dots X_{j_{\ell'}}^{(t_1+1)}, \dots, X_{j_1}^{(t)} \dots X_{j_{\ell'}}^{(t)}$ .

Defining good blocks (for a specific source, and globally good blocks) as in the first stage (albeit for the last  $t_2$  sources), we infer that (with very high probability) each cell is assigned at least  $((\delta/4)^2 - o(1)) \cdot \ell$  good blocks, since at least  $\delta/4$  of the assignments are random and the density of good blocks is at least  $\delta/4$ . (Recall that a block is good if it is good for at least a  $\delta/4$  fraction of the last  $t_2$  sources, and that being good for a source means having conditional min-entropy rate at least  $\delta/3$ .) In this case (i.e., for a typical assignment of blocks to cells), the average min-entropy rate of the blocks assigned to the smallest cell is at least  $((\delta/4)^2 - o(1)) \cdot \delta/3 = \Omega(\delta^3)$ , where the average is taken over the last  $t_2$  sources. It follows that  $\Omega(\delta^3)$  of the corresponding sub-sources (i.e., the sources restricted to the blocks assigned to a cell) have min-entropy rate of  $\Omega(\delta^3)$ . Applying Claim 7.7 (this time with  $\rho = \Theta(\delta^3)$  and again with  $h = \Theta(\log(1/\rho))$ ), we infer that the final output is close to uniform.

Finally, note that all operations (including the arithmetics in the finite fields, which have size  $\exp(\text{poly}(\log n))$ ) can be implemented by constant-depth circuits of size  $\text{poly}(n)$  (see [9, 50]).<sup>40</sup> ◀

**Trading-off sources for error**

Following Barak *et al.* [8], we can reduce the number of sources to a fixed constant (i.e., five) independent of the constant min-entropy rate of these sources, but this comes at the expense of increasing the extraction error to a larger  $o(1)$ .

► **Theorem 7.8** (deterministic extractor in  $\mathcal{AC}^0$  for five sources of constant rate  $\delta$ ). *For any constants  $\delta > 0$  there exists a uniform  $\mathcal{AC}^0$  function  $E : (\{0, 1\}^n)^5 \rightarrow \{0, 1\}^{\omega(1)}$  that constitutes a  $((\delta n, \delta n, \delta n, \delta n, \delta n), o(1))$ -extractor.*

<sup>39</sup> Recall that the min-entropy of  $g(X)g(Y) + g(Z)$  is at least the maximum of the min-entropy of these three (independent) random variables, where we use the fact that  $\Pr[g(X)=0] = 0$  (and ditto for  $Y$ ).

<sup>40</sup> Alternatively, one can use constructions over finite fields of characteristic two, and use the bounds stated in [Sec. 2.8]Z4 rather than [7, Lem. 3.14].



Indeed, the three-source extractor in [8, Thm. 1.1] is only claimed to extract a constant number of bits with constant error, but the proof (as is) establishes functions of the form  $\log^{\Omega(1)} n$  and  $\log^{-\Omega(1)} n$ , respectively. Our result just inherits these bounds.

**Proof Sketch.** We follow the strategy of the proof of Theorem 7.6, while implementing the two stages of extraction in a different manner. Specifically, the first extraction stage (which used the  $t_1$ -sources extractor  $E'$ ) will be performed by the two-source *somewhere extractor* of [8, Thm. 6.2], whereas the second extraction stage (which used the  $t_2$ -sources extractor  $E''$ ) will be performed by the three-source extractor of [8, Thm. 1.1]. These two extractors will be applied to  $\text{poly}(\log n)$ -bit long blocks of the various sources, and they can be implemented by explicit constant-depth  $\text{poly}(n)$ -size circuits, since the computations involved reduce to applications of the three-source extractor  $E_3$  of [7] and the bilinear extractor of [17, 20].<sup>41</sup>

Note that the two-source somewhere extractor of [8, Thm. 6.2] outputs a constant number of strings such that one of them is random (or rather an output-dependent choice of an element in the output sequence yields a distribution that is close to being uniform). Nevertheless, if we assign to cell  $c$  all blocks with index  $b$  such that  $(b, c)$  appeared in the output sequence of some invocation of the two-source somewhere extractor, then the analysis of the first stage remains valid, except that now the total size of the cells is a constant factor bigger. (Indeed, if  $E' : \{0, 1\}^\ell \times \{0, 1\}^\ell \rightarrow ([n'] \times [n'/\ell])^{O(1)}$  is the two-source somewhere extractor in use, then we use the  $O(1)$ -long output sequence  $E'(X_j^{(1)}, X_j^{(2)})$  for each  $j \in [n']$ .)

In the second stage, we shall use a three-source extractor  $E''$  that can handle min-entropy rate that is a constant factor smaller than the bound used in the proof of Theorem 7.6. Note that this extractor only outputs a small number of bits with an  $o(1)$  deviation from the uniform distribution.

In the analysis of both stages, we invoke Claim 7.3 and infer that in *each source* at least  $\delta/2$  of the blocks have (conditional) min-entropy rate of at least  $\delta/3$ . The problem is that, in order for extraction to work, we need (many) blocks that have sufficient min-entropy in *each of* the first two sources (resp., *each of* the last three sources). This problem is solved for the first two sources by using  $\text{poly}(1/\delta)$  matchings over  $[n']$ , rather than the single matching  $\{(j, j)\}_{j \in [n']}$ . Specifically, we use (partial) matchings that correspond to a partition of the edges of an expander graph into partial monotone functions. Such expanders are called monotone [21] and were constructed in [14]. Monotonicity is important here because the definition of a good block refers to the conditional min-entropy of the block, which in turn refers to a fixed order.

In light of the above, the first extraction stage is actually as follows. Let  $f_1, \dots, f_t : [n'] \rightarrow [n']$  be partial monotone functions such that  $t = \text{poly}(1/\delta)$  and for any set  $S$  of density  $\delta/2$  it holds that  $\cup_{i \in [t]} f_i(S)$  has density at least  $1 - \delta/4$ . (The second condition follows by applying the expander mixing lemma to the graph  $([n'], \cup_{i \in [t]} \{(v, f_i(v)) : v \in [n']\})$ .<sup>42</sup> For every  $j \in [n']$  and  $i \in [t]$ , where  $t' = \text{poly}(1/\delta)$  is the length of the sequence output by  $E' : \{0, 1\}^\ell \times \{0, 1\}^\ell \rightarrow ([n'] \times [n'/\log n])^{t'}$ , we compute the  $t'$ -long sequence  $E'(X_j^{(1)}, X_{f_i(j)}^{(2)})$ , and assign block  $b$  to cell  $c$  if  $(b, c)$  appeared in that sequence.

Now, letting  $B_1$  (resp.,  $B_2$ ) denote the set of blocks that are good for the first (resp., second) source, it follows that  $|B_1|, |B_2| \geq \delta n'/2$ . Hence,  $(\cup_{i \in [t]} f_i(B_1)) \cap B_2$  has density at

<sup>41</sup> For the second extractor (i.e., the three-source extractor of [8, Thm. 1.1]) the corresponding construction in [8, Thm. 1.1] also apply an optimal two-source extractor, but it is applied on strings of very short length.

<sup>42</sup> If  $f_i$  is undefined on  $v$ , then  $f_i(v)$  is ignored in the various expressions; for example  $f_i(S \cup \{v\}) = f_i(S)$  and  $(v, f_i(v)) \cup W = W$ .

least  $\delta/4$ , and there exists  $i \in [t]$  such that  $\{(j, f_i(j)) : j \in B_1 \wedge f_i(j) \in B_2\}$  has density at least  $\delta/4t$ . For this  $i$  and for every  $j \in B_1 \cap f_i^{-1}(B_2)$ , it holds that the  $j^{\text{th}}$  block in the first source (resp., the  $f_i(j)^{\text{th}}$  block in the second source) has min-entropy rate at least  $\delta/3$  conditioned on the prior blocks, where the blocks are ordered according to  $j$  (and we use  $f_i(j') < f_i(j)$  for  $j' < j$ ). Hence, the corresponding assignment (based on extraction from these two blocks (i.e.,  $X_j^{(1)}$  and  $X_{f_i(j)}^{(2)}$ )) will be *somewhere random*, conditioned on the prior assignment. It follows that at least  $\frac{\delta/4t}{t} = \text{poly}(\delta)$  fraction of the blocks will be assigned at random, which guarantees that (w.v.h.p.) every cell will be assigned at least  $\text{poly}(\delta) \cdot \ell$  random blocks. (Recall that extraction from at least  $\delta/4t$  blocks will be *somewhere random*, which essentially means that one of the  $t'$  extracted outputs is random.)

In the second extraction stage we shall also use matchings of the blocks of one source with blocks of the second source, but here we need to match the blocks of one source with blocks of two other sources, which can be done just in the same manner. (Indeed, we shall use a Monotone expander for density  $\delta/2$  and monotone degree  $t = \text{poly}(\delta)$  to determine the matching of blocks of  $X^{(3)}$  and blocks of  $X^{(4)}$  (as done for  $X^{(1)}$  and  $X^{(2)}$ ), but we shall use a Monotone expander for density  $\delta/4t$  and monotone degree  $t'' = \text{poly}(\delta/t)$  to determine the matching of blocks of  $X^{(3)}$  and blocks of  $X^{(5)}$ .) ◀

## 8 Open Problems (collected and restated)

In this section we collect and restate open problems that are mentioned in various parts of the paper. We start with problems regarding general min-entropy sources.

### Extraction from general min-entropy sources

The main open problem was stated as Problem 1.6. It refers to extracting more than poly-logarithmically many bits from a source of constant (or even  $1/\text{poly}(\log n)$ ) min-entropy rate using a logarithmically long seed. More generally, we have –

► **Open Problem 8.1** (Problem 1.6, restated). *Can extractors computable in  $\mathcal{AC}^0$  achieve an extraction rate that is greater than a poly-logarithmic function (i.e.,  $m(n)/r(n) > \text{poly}(\log n)$ ) when the min-entropy rate is at least  $1/\text{poly}(\log n)$  and  $r(n) = \Omega(\log n)$ ?*

Recall that for  $k(n) \geq n/\text{poly}(\log n)$  and  $r(n) = O(\log n)$ , we have  $\mathcal{AC}^0$ -extractors either for the case of  $m(n) = r(n) + \Omega(\log n)$  and  $\epsilon(n) = 1/\text{poly}(n)$  (see Part 1 of Corollary 3.6) or for the case of  $m(n) = \text{poly}(\log n)$  and  $\epsilon(n) = 1/\text{poly}(\log n)$  (see Part 2 of Corollary 3.6). We left open the following problem.

► **Open Problem 8.2** (closing a gap w.r.t the error parameter). *For  $k(n) \geq n/\text{poly}(\log n)$  and  $r(n) = O(\log n)$ , can  $\mathcal{AC}^0$ -extractors extract more than logarithmically many bits with  $1/\text{poly}(n)$  error? Moreover, for any polynomial  $p$ , can they extract  $p(\log n)$  bits with error  $\epsilon(n) = 1/p(n)$ ?*

Turning back to Problem 8.1, recall that a positive resolution of Problem 4.8 (which refers to converting general sources to block sources) is a sufficient but unnecessary condition for a positive resolution of a version of Problem 8.1 (when both are qualified with respect to constant min-entropy rate).

► **Open Problem 8.3** (Problem 4.8, restated). *Does there exist a  $(\delta, \delta', o(1))$ -blockers  $S : \{0, 1\}^{O(\log n)} \rightarrow ([n]^s)^m$  for constants  $\delta \in (0, 1)$  and  $\delta' \in (0, \delta)$  and some unbounded  $m = \omega(1)$ ? What about  $m = n^{\Omega(1)}$  and  $s = \text{poly}(\log n)$ ? And what about  $\delta \geq 1/\text{poly}(\log n)$  and  $\delta - \delta' \geq 1/\text{poly}(\log n)$ ?*

### Extraction from restricted sources

While we proved the existence of deterministic  $\mathcal{AC}^0$ -extractors for bit-fixing sources and for pairs of independent sources, our results in these cases have deficiencies.

Firstly, in the two-source model we only obtained deterministic  $\mathcal{AC}^0$ -extractors for entropy rate that tends to 1 (see Theorem 7.2). Recall that non-explicit functions (outside of  $\mathcal{AC}^0$ ) can extract from pairs of sources of logarithmic min-entropy [17, Thm. 7], but the impossibility results in Section 5.2 can be adapted to rule out min-entropy rates below  $1/\text{poly}(\log n)$ . On the other hand, above that level of min-entropy  $\mathcal{AC}^0$ -extraction *with logarithmic seed length* is possible. Hence, it is reasonable to ask whether deterministic  $\mathcal{AC}^0$ -extractors can meet this performance (as in the case of bit-fixing sources).

► **Open Problem 8.4** (two-source deterministic extraction in  $\mathcal{AC}^0$ ). *Does there exist a  $((k, k), \epsilon)$ -extractor in  $\mathcal{AC}^0$  (even extracting just a single bit) for any  $k(n) = n/\text{poly}(\log n)$  and  $\epsilon(n) = 1/\text{poly}(n)$ ?*

For starters, one may consider the case of constant min-entropy rate; that is, *is any  $(\Omega(n), \Omega(n), \Omega(1))$ -extractor computable in  $\mathcal{AC}^0$ ?* Recall that  $\mathcal{AC}^0$  circuits can extract from four sources of some constant (i.e., 0.99) min-entropy rate (cf. Theorem 7.5), and they can extract from  $\text{poly}(1/\delta)$ -many sources of any constant min-entropy rate  $\delta > 0$  (cf. Theorem 7.6).

We mention that, as shown by Chor and Goldreich [17, Sec. 4], any function that is a  $(\Omega(n), \Omega(n), \Omega(1))$ -extractor has  $\Omega(n)$  distributional communication complexity *with respect to the uniform distribution*. Whether such a function exists in  $\mathcal{AC}^0$  is open. (Note that the  $\Omega(n)$  distributional communication complexity of set disjointness is not with respect to the uniform distribution.)

A second deficiency in the aforementioned results regarding deterministic extractors is that these  $\mathcal{AC}^0$ -extractors are non-explicit. Hence, we ask for explicit versions of these results. Since all results for the bit-fixing source model are obtained by uniform  $\mathcal{AC}^0$ -reductions from Theorem 5.8, it suffices to have an explicit version of the latter.

► **Open Problem 8.5** (explicit version of Theorem 5.8 – deterministic extraction in uniform  $\mathcal{AC}^0$  for bit-fixing sources). *For every  $k(n) \geq n/\text{poly}(\log n)$  and every  $\epsilon(n) > 1/\text{poly}(\log n)$ , present deterministic  $\epsilon$ -error extractors  $E : \{0, 1\}^n \rightarrow \{0, 1\}$  for  $(n, k)$ -bit-fixing sources such that the extractors are computable in uniform  $\mathcal{AC}^0$ .*

For extraction from pairs of sources, our first challenge is to provide an explicit construction that matches the parameters of Theorem 7.2. However, since we believe that Theorem 7.2 can be improved w.r.t the required min-entropy rate, we state a more general challenge.

► **Open Problem 8.6** (two-source deterministic extraction in uniform  $\mathcal{AC}^0$ ). *For every  $k, \epsilon$  such that the existence of  $((k, k), \epsilon)$ -extractor in  $\mathcal{AC}^0$  is established, provide an explicit construction (i.e., a construction in uniform  $\mathcal{AC}^0$ ).*

An intermediate step towards resolving Problems 8.5 and 8.6 is providing an explicit construction of a deterministic extractor for *non-oblivious* bit-fixing sources with parameters matching those in Theorem 5.7. Such an explicit construction would be of independent interest. Actually, for our current applications, it will suffice to establish the following:

► **Open Problem 8.7** (explicit version of Theorem 5.7 – deterministic extraction in uniform  $\mathcal{AC}^0$  for non-oblivious bit-fixing sources). *For some  $\rho = 1/\text{poly}(\log n)$ , provide uniform  $\mathcal{AC}^0$  circuits  $C : \{0, 1\}^n \rightarrow \{0, 1\}$  such that  $C$  is balanced and the influence of every set of density  $\rho$  on  $C$  is at most 0.1.*

Recall that the non-explicit circuits of Ajtai and Linial [4] (i.e., Theorem 5.7) support  $\rho = 1/O(\log n)^2$ .

**Acknowledgments.** We are grateful to Ronen Shaltiel and Salil Vadhan for useful discussions. O.G. was partially supported by the Minerva Foundation with funds from the Federal German Ministry for Education and Research. E.V. was supported by NSF grant CCF-1319206. Work done in part while E.V. was a visiting scholar at Harvard University, with support from Salil Vadhan's Simons Investigator grant. A.W. was partially supported by NSF grant CCF-1412958.

---

## References

- 1 Miklós Ajtai.  $\Sigma_1^1$ -formulae on finite structures. *Annals of Pure and Applied Logic*, 24(1):1–48, 1983.
- 2 Miklós Ajtai. Approximate counting with uniform constant-depth circuits. In *Advances in computational complexity theory*, pages 1–20. Amer. Math. Soc., Providence, RI, 1993.
- 3 Miklos Ajtai and Michael Ben-Or. A Theorem on Probabilistic Constant Depth Computations. In *16th ACM Symp. on the Theory of Computing*, 471–474, 1984.
- 4 Miklos Ajtai and Nathan Linial. The influence of large coalitions. *Combinatorica*, Vol. 13 (2), pages 129–145, 1993.
- 5 Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. On Pseudorandom Generators with Linear Stretch in NC. *Computational Complexity*, Vol. 17(1), pages 38–69, 2008. Preliminary version in *10th RANDOM*, 2006.
- 6 László Babai. Random oracles separate PSPACE from the polynomial-time hierarchy. *Information Processing Letters*, 26(1):51–53, 1987.
- 7 Boaz Barak, Russell Impagliazzo, and Avi Wigderson. Extracting Randomness Using Few Independent Sources. *SIAM Journal on Computing*, Vol. 36(4), pages 1095–1118, 2006. Preliminary version in *45th FOCS*, 2004.
- 8 Boaz Barak, Guy Kindler, Ronen Shaltiel, Benny Sudakov, and Avi Wigderson. Simulating independence: new constructions of condensers, ramsey graphs, dispersers, and extractors. *Journal of the ACM*, Vol. 57(4), 2010. Preliminary version in *37th STOC*, 2005.
- 9 Paul Beame, Stephen Cook, and James Hoover. Log-Depth Circuits for Division and Related Problems. *SIAM Journal on Computing*, Vol. 15, pages 994–1003, 1986.
- 10 Johannes Blomer, Richard M. Karp and Emo Welzl. The rank of sparse random matrices over finite fields. *Random Struct. Algorithms*, Vol. 10(4), pages 407–419, 1997.
- 11 Andrej Bogdanov and Siyao Guo. Sparse extractor families for all the entropy. In *ACM Innovations in Theoretical Computer Science conf. (ITCS)*, pages 553–560, 2013.
- 12 Ravi B. Boppana. The Average Sensitivity of Bounded-Depth Circuits. *Information Processing Letters*, 63(5), pages 257–261, 1997.
- 13 Jean Bourgain. More on the Sum-Product Phenomenon in Prime Fields and its Applications. *Int. J. Number Theory*, Vol. 1, pages 1–32, 2005.
- 14 Jean Bourgain and Amir Yehudayoff. Monotone Expansion. In *44th ACM Symp. on the Theory of Computing*, pages 1061–1078, 2012.
- 15 Larry Carter and Mark N. Wegman. Universal Hash Functions. *Journal of Computer and System Science*, Vol. 18, pages 143–154, 1979.
- 16 Shiva Chaudhuri and Jaikumar Radhakrishnan. Deterministic restrictions in circuit complexity. In *28th ACM Symp. on the Theory of Computing*, pages 30–36, 1996.
- 17 Benny Chor and Oded Goldreich. Unbiased Bits from Sources of Weak Randomness and Probabilistic Communication Complexity. *SIAM Journal on Computing*, Vol. 17(2), pages 230–261, 1988. Preliminary version in *26th FOCS*, 1985.
- 18 Benny Chor, Joel Friedman, Oded Goldreich, Johan Hastad, Steven Rudich, and Roman Smolensky. The Bit Extraction Problem or t-Resilient Functions. In *26th IEEE Symp. on Foundations of Computer Science*, pages 396–407, 1985.

- 19 Gil Cohen and Igor Shinkar. Zero-Fixing Extractors for Sub-Logarithmic Entropy. *ECCC*, TR14-160, 2014.
- 20 Yevgeniy Dodis, Ariel Elbaz, Roberto Oliveira, and Ran Raz. Improved Randomness Extraction from Two Independent Sources. In *8th RANDOM*, Springer LNCS, pages 334–344, 2004.
- 21 Zeev Dvir and Avi Wigderson. Monotone Expanders: Constructions and Applications. *Theory of Computing*, Vol. 6(1), pages 291–308, 2010.
- 22 Uriel Feige. Noncryptographic Selection Protocols. In *40th IEEE Symp. on Foundations of Computer Science*, pages 142–153, 1999.
- 23 Ehud Friedgut. Influences in Product Spaces: KKL and BKKKL Revisited. *Comb., Prob. & Comp.*, Vol. 13(1), pages 17–29, 2004.
- 24 Merrick L. Furst, James B. Saxe, and Michael Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical Systems Theory*, 17(1):13–27, 1984.
- 25 Ariel Gabizon and Ran Raz. Deterministic extractors for affine sources over large fields. *Combinatorica*, Vol. 28 (4), pages 415–440, 2008. Preliminary version in *46th FOCS*, 2005.
- 26 Ariel Gabizon, Ran Raz, and Ronen Shaltiel. Deterministic Extractors for Bit-Fixing Sources by Obtaining an Independent Seed. *SIAM Journal on Computing*, Vol. 36 (4), pages 1072–1094, 2006. Preliminary version in *45th FOCS*, 2004.
- 27 Oded Goldreich. *Computational Complexity: A Conceptual Perspective*. Cambridge University Press, 2008.
- 28 Oded Goldreich. A Sample of Samplers: A Computational Perspective on Sampling. In *Studies in Complexity and Cryptography*. Springer LNCS 6650, pages 302–332, 2011.
- 29 Oded Goldreich and Avi Wigderson. Tiny families of functions with random properties: A quality-size trade-off for hashing. *Random Structures and Algorithms*, Vol. 11(4), pages 315–343, 1997. Preliminary version in *26th STOC*, 1994.
- 30 Oded Goldreich and Avi Wigderson. Derandomizing algorithms that err extremely rarely. In *46th ACM Symp. on the Theory of Computing*, pages 109–118, 2014. Full version available from *ECCC*, TR13-152, 2013.
- 31 Venkatesan Guruswami, Christopher Umans, and Salil P. Vadhan. Unbalanced expanders and randomness extractors from Parvaresh–Vardy codes. *Journal of the ACM*, Vol. 56(4), 2009. Preliminary version in *ECCC*, TR06-134, 2006.
- 32 Johan Hastad. Almost Optimal Lower Bounds for Small Depth Circuits. *Advances in Computing Research: a research annual*, Vol. 5 (Randomness and Computation, S. Micali, ed.), pages 143–170, 1989. Extended abstract in *18th STOC*, 1986.
- 33 Alexander Healy and Emanuele Viola. Constant-Depth Circuits for Arithmetic in Finite Fields of Characteristic Two. In *STACS*, pages 672–683, 2006. See also *ECCC*, TR05-087, 2005.
- 34 Russell Impagliazzo and Moni Naor. Efficient cryptographic schemes provably as secure as subset sum. *J. of Cryptology*, 9(4):199–216, 1996.
- 35 Russell Impagliazzo, Ramamohan Paturi, and Michael E. Saks. Size-Depth Tradeoffs for Threshold Circuits. *SIAM Journal on Computing*, Vol. 26(3), pages 693–707, 1997. Preliminary version in *25th STOC*, 1993.
- 36 Russell Impagliazzo and Avi Wigderson. P=BPP if E requires exponential circuits: Derandomizing the XOR Lemma. In *29th ACM Symp. on the Theory of Computing*, pages 220–229, 1997.
- 37 Statsys Jukna. *Boolean Function Complexity: Advances and Frontiers*. Algorithms and Combinatorics, Vol. 27, Springer, 2012.
- 38 Xin Li. Extractors for a Constant Number of Independent Sources with Polylogarithmic Min-Entropy. In *54th IEEE Symp. on Foundations of Computer Science*, pp. 100–109, 2013.



- 39 Nathan Linial, Yishay Mansour, and Noam Nisan. Constant Depth Circuits, Fourier Transform, and Learnability. *Journal of the ACM*, Vol. 40(3), pages 607–620, 1993. Preliminary version in *30th FOCS*, 1989.
- 40 Jeff Kahn, Gil Kalai, and Nathan Linial. The Influence of Variables on Boolean Functions (Extended Abstract). In *29th IEEE Symp. on Foundations of Computer Science*, pages 68–80, 1988.
- 41 Jesse Kamp and David Zuckerman. Deterministic Extractors for Bit-Fixing Sources and Exposure-Resilient Cryptography. *SIAM Journal on Computing*, Vo. 36(5), pages 1231–1247, 2007. Preliminary version in *44th FOCS*, 2003.
- 42 Alexander Lubotzky, Ralph Phillips, and Peter Sarnak. Ramanujan graphs. *Combinatorica*, Vol. 8(3), pages 261–277, 1988. Preliminary version in *18th STOC*, 1986.
- 43 Yishay Mansour, Noam Nisan, and Prason Tiwari. The Computational Complexity of Universal Hashing. *Theor. Comput. Sci.*, Vol. 107(1), pages 121–133, 1993. Preliminary version in *22nd STOC*, 1990.
- 44 Noam Nisan. Pseudorandom bits for constant depth circuits. *Combinatorica*, Vol. 11 (1), pages 63–70, 1991.
- 45 Noam Nisan and Avi Wigderson. Hardness vs Randomness. *Journal of Computer and System Science*, Vol. 49, No. 2, pages 149–167, 1994. Preliminary version in *29th FOCS*, 1988.
- 46 Noam Nisan and David Zuckerman. Randomness is Linear in Space. *Journal of Computer and System Science*, 52(1):43–52, 1996. Preliminary version in *25th STOC*, 1993.
- 47 Ryan O’Donnell. *Analysis of Boolean Functions Hardcover*. Cambridge University Press, 2014.
- 48 Prabhakar Ragde and Avi Wigderson. Linear-Size Constant-Depth Polylog-Threshold Circuits. *Information Processing Letters*, Vol. 39(3), pages 143–146, 1991.
- 49 Ran Raz, Omer Reingold, Salil P. Vadhan. Error Reduction for Extractors. In *40th IEEE Symp. on Foundations of Computer Science*, pages 191–201, 1999.
- 50 John Reif. On Threshold Circuits and Polynomial Computation. In *2nd Conf. on Structure in Complexity Theory*, pages 118–123, 1987.
- 51 Omer Reingold, Ronen Shaltiel, and Avi Wigderson. Extracting Randomness via Repeated Condensing. *SIAM Journal on Computing*, Vol. 35 (5), pages 1185–1209, 2006. Preliminary version in *41st FOCS*, 2000.
- 52 Miklos Santha and Umesh V. Vazirani. Generating Quasi-random Sequences from Semi-random Sources. *Journal of Computer and System Science*, Vol. 33(1), pages 75–87 (1986). Preliminary version in *25th FOCS*, 1984.
- 53 Ronen Shaltiel. Recent Developments in Explicit Constructions of Extractors. In *Current Trends in Theoretical Computer Science: The Challenge of the New Century, Vol 1: Algorithms and Complexity*, World scietific, 2004. (Editors: G. Paun, G. Rozenberg and A. Salomaa.) Preliminary version in *Bulletin of the EATCS 77*, pages 67–95, 2002.
- 54 Luca Trevisan. Extractors and Pseudorandom Generators. *Journal of the ACM*, Vol. 48 (4), pages 860–879, 2001. Preliminary version in *31st STOC*, 1999.
- 55 Salil P. Vadhan. Constructing Locally Computable Extractors and Cryptosystems in the Bounded-Storage Model. *Journal of Cryptology*, Vol. 17(1), pages 43–77, 2004.
- 56 Umesh V. Vazirani. Towards a Strong Communication Complexity Theory or Generating Quasi-Random Sequences from Two Communicating Slightly-random Sources (Extended Abstract). In the *17th ACM Symp. on the Theory of Computing*, pages 366–378, 1985.
- 57 Emanuele Viola. The complexity of constructing pseudorandom generators from hard functions. *Computational Complexity*, Vol. 13 (3-4), pages 147–188, 2005. Preliminary version in *18th CCC*, 2003.

- 58 Emanuele Viola. On Constructing Parallel Pseudorandom Generators from One-Way Functions. In the *20th IEEE Conf. on Computational Complexity*, pages 183–197, 2005.
- 59 Emanuele Viola. On Approximate Majority and Probabilistic Time. *Computational Complexity*, Vol. 18 (3), pages 337–375, 2009. Preliminary version in *22nd CCC*, 2007.
- 60 Andrew Yao. Separating the polynomial-time hierarchy by oracles. In *26th IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 1–10, 1985.
- 61 David Zuckerman. General Weak Random Sources. In *31st IEEE Symp. on Foundations of Computer Science*, pages 534–543, 1990.
- 62 David Zuckerman. Simulating BPP Using a General Weak Random Source. In *32nd IEEE Symp. on Foundations of Computer Science*, pages 79–89, 1991.
- 63 David Zuckerman. Randomness-optimal oblivious sampling. *Random Structures and Algorithms*, Vol. 11(4), pages 345–367, 1997. Preliminary version in *28th STOC*, 1996.
- 64 David Zuckerman. Linear Degree Extractors and the Inapproximability of Max Clique and Chromatic Number. *Theory of Computing*, Vol. 3, pages 103–128, 2007.

## A

 Appendices

Appendix A.1 presents some observations made in passing, regarding a notion used in this work. In contrast, Appendix A.2 presents a known (but not well-known) result that is used in this work. Lastly, Appendix A.3 presents a somewhat different solution to the problem of counting upto  $\text{poly}(\log n)$  in  $\mathcal{AC}^0$ .

### A.1 On the robustness of averaging samplers

We prove two “robustness claims” regarding the notion of (relaxed) averaging samplers (as in Definition 2.6). These claims are not essential to the current write-up, but we consider it worthy to present them. The first claim asserts that samplers that perform well for Boolean functions also perform well for general functions.

► **Definition A.1** (Boolean averaging samplers, a restriction of Definition 2.6). A function  $S : \{0, 1\}^r \rightarrow [n]^t$  is called a Boolean  $(\mu, \mu', \gamma)$ -averaging sampler if Eq. (1) holds for every  $f : [n] \rightarrow \{0, 1\}$  such that  $\rho(f) \stackrel{\text{def}}{=} \mathbf{E}_{i \in [n]} [f(i)] \geq \mu$ .

The aforementioned claim, stated and proved next, is analogous to the corresponding claim for standard averaging samplers (cf. [28, Thm. 5.10]).

► **Claim A.2** (Boolean vs general averaging samplers). Let  $\mu, \mu', \epsilon \in (0, 1]$  and  $n > t = \Omega(\mu\epsilon^{-2} \log(1/\gamma))$ . If  $S : \{0, 1\}^r \rightarrow [n]^t$  is a Boolean  $(\mu, \mu', \gamma)$ -averaging sampler, then  $S$  is a  $(\mu + \epsilon, \mu' - \epsilon, 3\gamma)$ -averaging sampler.

The hypothesis  $n > t = \Omega(\mu\epsilon^{-2} \log(1/\gamma))$  is not a real restriction when  $\epsilon = \Omega(\mu - \mu')$ , since any  $(\mu, \mu', \gamma)$ -sampler must satisfy it (cf., e.g., the discussion in [28]).<sup>43</sup>

**Proof.** The proof mimics the proof of the corresponding claim for standard averaging samplers (cf. [28, Thm. 5.10]). For any function  $f : [n] \rightarrow [0, 1]$  such that  $\rho(f) \geq \mu + \epsilon$ , we consider a mental experiment in which the sampler is invoked on a random Boolean function that reflects  $f$ . Specifically, we consider a random Boolean function  $b : [n] \rightarrow \{0, 1\}$  such that  $b(i) = 1$  with probability  $f(i)$  and  $b(i) = 0$  otherwise, for every  $i \in [n]$  and independently of

<sup>43</sup> Indeed, one alternative ending of the proof of Claim 3.4 uses Claim A.2; in that application,  $\mu$  and  $\mu'$  are both constants in  $(0, 1)$ , and  $\epsilon = (1 - \mu\mu')/2$ .



the setting of all other values. Then, with probability at least  $1 - \exp(-(\epsilon/\mu)^2 \cdot \mu n) > 1 - \gamma$  (over the choice of  $b$ ), it holds that  $\rho(b) \geq \rho(f) - \epsilon \geq \mu$ . In this case, with probability at least  $1 - \gamma$  (over the choice of  $I$  when  $b$  is fixed arbitrarily), the sample  $I$  chosen by the Boolean averaging sampler satisfies  $\sum_{i \in I} b(i) \geq \mu'$ . Lastly, with probability at least  $1 - \exp(-(\epsilon/\mu)^2 \cdot \mu t) > 1 - \gamma$  (over the choice of  $b$  when  $I$  is fixed arbitrarily), it holds that  $\sum_{i \in I} f(i) \geq \sum_{i \in S} b(i) - \epsilon \geq \mu' - \epsilon$ . The claim follows. ◀

The second claim asserts an upwards translation of the performance guarantee of (relaxed) averaging samplers. Note that the translation (from  $(\mu, \mu', \gamma)$  to  $(m \cdot \mu, m \cdot \mu', m \cdot \gamma)$ ) preserves the relative error (i.e.,  $(\mu - \mu')/\mu$ ) rather than the absolute error (i.e.,  $\mu - \mu'$ ).

► **Claim A.3** (upward translation of the performance guarantee). *If  $S : \{0, 1\}^r \rightarrow [n]^t$  is a  $(\mu, \mu', \gamma)$ -averaging sampler, then for every  $m \in \mathbb{N}$  it holds that  $S$  is a  $(m \cdot \mu, m \cdot \mu', m \cdot \gamma)$ -averaging sampler. The same holds with respect to Boolean averaging samplers.*

The claim holds trivially for any  $(\mu, \mu', \gamma)$ -averaging sampler that is actually a standard averaging sampler (i.e., one that approximates the average value of any function up to an addition term of  $\mu - \mu'$  with error probability  $1 - \gamma$ ).

**Proof.** The claim is proved by considering auxiliary functions that share the “weight” of the target function  $f$  such that each auxiliary function takes approximately an equal share of the weight of  $f$ . Specifically, given  $f : [n] \rightarrow [0, 1]$  such that  $\mathbf{E}_{i \in [n]}[f(i)] \geq m \cdot \mu$ , consider  $m$  auxiliary functions  $f_j : [n] \rightarrow [0, 1]$  such that  $f(i) = \sum_{j \in [m]} f_j(i)$  and  $\mathbf{E}_{i \in [n]}[f_j(i)] \geq \mu$  for every  $j \in [m]$ . Note that if  $f$  is Boolean then (w.l.o.g.) so are the  $f_j$ 's. Now, by Eq. (1), for every  $j \in [m]$  it holds that

$$\Pr_{I \leftarrow S(U_r)} \left[ \frac{1}{t} \sum_{i \in I} f_j(i) < \mu' \right] \leq \gamma$$

and the claim follows by a union bound. ◀

## A.2 A standard high moment inequality

The following concentration bound for somewhat independent random variables is well known to the experts, but is hard to find in standard texts.

► **Lemma A.4** (folklore). *Let  $\zeta_1, \dots, \zeta_n$  be identical random variables that are distributed in  $[0, 1]$  in a  $2k$ -wise independent manner, and let  $M = \mathbf{E}[\sum_{i \in [n]} \zeta_i]$ . Then, for any  $\epsilon > 0$ , it holds that*

$$\Pr \left[ \left| \sum_{i \in [n]} \zeta_i - M \right| > \epsilon \cdot M \right] < (\epsilon^{-2} k^2 / M)^k$$

**Proof.** Let  $\bar{\zeta}_i = \zeta_i - \mathbf{E}[\zeta_i]$  and note that  $\mathbf{E}[\bar{\zeta}_i] = 0$ . By Markov's inequality and linearity of expectation, we have

$$\begin{aligned} \Pr \left[ \left| \sum_{i \in [n]} \zeta_i - M \right| > \epsilon \cdot M \right] &\leq \frac{\mathbf{E} \left[ \left( \sum_{i \in [n]} \bar{\zeta}_i \right)^{2k} \right]}{(\epsilon \cdot M)^{2k}} \\ &= \frac{\sum_{i_1, \dots, i_{2k} \in [n]} \mathbf{E} \left[ \prod_{j \in [2k]} \bar{\zeta}_{i_j} \right]}{(\epsilon \cdot M)^{2k}} \end{aligned} \tag{26}$$

Now, the key observation is that terms in which some variable appears exactly once do not contribute to the sum in Eq. (26). In general, by virtue of  $2k$ -wise independence, for any  $t \leq 2k$ , a term in which variables indexed  $j_1, \dots, j_t$  occur with multiplicities  $e_1, \dots, e_t$  contributes  $\mathbf{E} \left[ \prod_{\ell \in [t]} \bar{\zeta}_{j_\ell}^{e_\ell} \right] = \prod_{\ell \in [t]} \mathbf{E}[\bar{\zeta}_{j_\ell}^{e_\ell}]$ . Denoting by  $S(n, 2k, j)$  the set of  $2k$ -long sequences over  $[n]$  in which exactly  $j$  variables appear and each of these variables appears with multiplicity at least two, we have

$$\begin{aligned} \sum_{i_1, \dots, i_{2k} \in [n]} \mathbf{E} \left[ \prod_{j \in [2k]} \bar{\zeta}_{i_j} \right] &= \sum_{j \in [k]} \sum_{(i_1, \dots, i_{2k}) \in S(n, 2k, j)} \mathbf{E} \left[ \prod_{j \in [2k]} \bar{\zeta}_{i_j} \right] \\ &\leq \sum_{j \in [k]} |S(n, 2k, j)| \cdot (M/n)^j \end{aligned}$$

where the inequality uses the fact that for every  $e \geq 2$  it holds that  $\mathbf{E}[\bar{\zeta}_i^e] \leq \mathbf{E}[\zeta_i^e] \leq \mathbf{E}[\zeta_i] = M/n$ . Using  $|S(n, 2k, j)| < \binom{n}{j} \cdot j^{2k} < n^j \cdot k^{2k}/2$ , we get

$$\begin{aligned} \Pr \left[ \left| \sum_{i \in [n]} \zeta_i - M \right| > \epsilon \cdot M \right] &< \frac{0.5k^{2k} \cdot \sum_{j \in [k]} n^j \cdot (M/n)^j}{(\epsilon \cdot M)^{2k}} \\ &< \frac{k^{2k}}{(\epsilon^2 \cdot M)^k} \end{aligned}$$

and the lemma follows. ◀

### A.3 Counting few ones in a long string

We consider the problem of counting the number of ones in a string, when guaranteed that this number is small. Specifically, for  $\ell = \text{poly}(\log n)$ , given an  $n$ -bit string  $x$  such that  $s \stackrel{\text{def}}{=} \sum_{i \in [n]} x_i \leq \ell$ , we seek explicit  $\mathcal{AC}^0$ -circuits that compute  $s$ . A solution to this problem has been known for decades; see [48], improving over [3, Sec. 5] (and subsequent works). For sake of elegance, we present a somewhat different solution here.

We use a small (i.e.,  $\text{poly}(n)$ -sized) family of pairwise independent hash functions mapping  $[n]$  to  $[\ell^2]$ . Such functions can be described by string of length  $2 \log n$ , and so they can be generically evaluated by depth-two circuits of  $\text{poly}(n)$  size. A random function in such family  $H$  shatters any set of size  $\ell$  with constant probability; that is, for every set  $I \subset [n]$  such that  $|I| \leq \ell$  it holds that  $\Pr_{h \in H}[|h(I)| = |I|] = \Omega(1)$  (since the collision probability is  $1/\ell^2$ ).

Now, on input  $x$  as above, we enumerate all  $h \in H$ , compute for every  $h$  the value of  $\sum_{v \in [\ell^2]} (\bigvee_{i \in h^{-1}(v)} x_i)$ , and take the maximum value. Indeed, if  $h$  shatters  $I = \{i : x_i = 1\}$ , then the value we computed equals  $\sum_{i \in [n]} x_i$ . The above computation is in  $\mathcal{AC}^0$  since we compute a  $\ell^2$ -wise sum (of some unbounded ors).<sup>44</sup>

---

<sup>44</sup>Indeed,  $\bigvee_{i \in h^{-1}(v)} x_i$  can be computed as  $\bigvee_{i \in [n]} (x_i \vee (h(i) = v))$ .