31st Conference on Computational Complexity

CCC'16, May 29 to June 1, 2016, Tokyo, Japan

Edited by

Ran Raz

Editor

Ran Raz
Department of Computer Science and Applied Math.
Weizmann Institute of Science
Rehovot 76100
Israel
ran.raz@weizmann.ac.il

ACM Classification 1998
F. Theory of Computation

ISBN 978-3-95977-008-8

Published online and open access by
Schloss Dagstuhl – Leibniz-Zentrum für Informatik GmbH, Dagstuhl Publishing, Saarbrücken/Wadern, Germany. Online available at http://www.dagstuhl.de/dagpub/978-3-95977-008-8.

Publication date
May, 2016

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.

Digital Object Identifier: 10.4230/10.4230/LIPIcs.CCC.2016.i

LIPIcs – Leibniz International Proceedings in Informatics

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

ISSN 1868-8969

http://www.dagstuhl.de/lipics

Contents

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
COMPUTATIONAL COMPLEXITY CONFERENCE

Preface

The papers in this volume were accepted for presentation at the 31st Conference on Computational Complexity (CCC'16), held May 29 to June 1, 2016 in Tokyo, Japan. The conference is organized by the Computational Complexity Foundation (CCF) and the Center for Exploring the Limits of Computation (CELC) in cooperation with the European Association for Theoretical Computer Science (EATCS) and the ACM Special Interest Group on Algorithms and Computation Theory (SIGACT). CCC'16 is sponsored by Microsoft Research.

The call for papers sought original research papers in all areas of computational complexity theory. Of the 91 submissions the program committee selected 34 for presentation at the conference.

The program committee would like to thank everyone involved in the conference, including all those who submitted papers for consideration as well as the reviewers for their scientific contributions; the board of trustees of the Computational Complexity Foundation and especially its president Dieter van Melkebeek for extensive advice and assistance; Jacobo Toran and Jeff Kinne for a variety of assistance; David Zuckerman for sharing his knowledge as 2015 PC chair; the Local Arrangements Committee and especially its chair Osamu Watanabe; Nisheeth Vishnoi for contributing an invited talk; and Marc Herbstritt for coordinating the production of these proceedings.

Ran Raz
Program Committee Chair

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Awards

The program committee of the 31st Conference on Computational Complexity is happy to present the *Best Paper Award* to Marco Carmosino, Russell Impagliazzo, Valentine Kabanets and Antonina Kolokolova for their paper

"Learning Algorithms from Natural Proofs".[1]

[1] See http://dx.doi.org/10.4230/LIPIcs.CCC.2016.34.

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Conference Organization

Program Committee

Anindya De, Northwestern University
Prahladh Harsha, Tata Institute of Fundamental Research
Neeraj Kayal, Microsoft Research
Jakob Nordström, KTH Royal Institute of Technology
Toniann Pitassi, University of Toronto
Anup Rao, University of Washington
Ran Raz (chair), Weizmann Institute & IAS
David Steurer, Cornell University
Thomas Vidick, California Institute of Technology
Amir Yehudayoff, Technion
Sergey Yekhanin, Microsoft Research

Local Arrangements Committee

Takashi Horiyama, Saitama University
Akinori Kawachi, Tokushima University
Takeshi Koshiba (co-chair), Saitama University
Kazuhisa Makino, Kyoto University
Ryuhei Mori, Tokyo Institute of Technology
Jun Tarui, University of Electro-Communications
Osamu Watanabe (chair), Tokyo Institute of Technology

Board of Trustees

Eric Allender (Treasurer), Rutgers University
Boaz Barak, Harvard University
Venkatesan Guruswami, Carnegie Mellon University
Jeff Kinne (Secretary), Indiana State University
Dieter van Melkebeek (President), University of Wisconsin-Madison
Madhu Sudan, Harvard University
Jacobo Toran, University of Ulm
Osamu Watanabe, Tokyo Institute of Technology

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
COMPUTATIONAL COMPLEXITY CONFERENCE

External Reviewers

Scott Aaronson
James Aisenberg
Arturs Backurs
Aleksandrs Belovs
Eric Blais
Michael Bremner
Siu On Chan
Ruiwen Chen
Daniel Dadush
Jan Elffers
Michael A. Forbes
Mika Göös
Rohit Gurjar
Samuel Hopkins
Rahul Jain
Yael Tauman Kalai
Antonina Kolokolova
Massimo Lauria
Troy Lee
Dieter Van Melkebeek
Jelani Nelson
Igor Carboni Oliveira
Youming Qiao
Ron Rothblum
Chandan Saha
Rahul Santhanam
Nitin Saxena
Ronen Shaltiel
Amir Shpilka
Noah Stephens-Davidowitz
Li-Yang Tan
Vinod Vaikuntanathan
Nisheeth Vishnoi
Omri Weinstein
Angela Wu

Divesh Aggarwal
Eric Allender
Paul Beame
Arnab Bhattacharyya
Andrej Bogdanov
Joshua Brody
Arkadev Chattopadhyay
Alessandro Chiesa
Susanna F. de Rezende
Uriel Feige
Benjamin Fuller
Joshua Grochow
Ronald de Haan
Pavel Hrubes
Stacey Jeffery
Tali Kaufman
Ashutosh Kumar
Francois Le Gall
Satya Lokam
Mladen Miksa
Huy Nguyen
Rafael Pass
Charles Rackoff
Aviad Rubinstein
Rishi Saket
Ramprasad Saptharishi
Rocco Servedio
Asaf Shapira
Florian Speelman
Xiaoming Sun
Sébastien Tavenas
Marc Vinyals
John Watrous
Ryan Williams
Grigory Yaroslavtsev

Nir Ailon
Per Austrin
Xiaohui Bei
Nir Bitansky
Ilario Bonacina
Clément Canonne
Eshan Chattopadhyay
Kai-Min Chung
Andrew Drucker
Yuval Filmus
Konstantinos Georgiou
Ankit Gupta
Pooya Hatami
Sangxia Huang
Valentine Kabanets
Hartmut Klauck
Mrinal Kumar
James Lee
Shachar Lovett
Moni Naor
Ryan O'Donnell
Sylvain Perifel
Jaikumar Radhakrishnan
Atri Rudra
Miklos Santha
Shubhangi Saraf
C. Seshadhri
Alexander Sherstov
Srikanth Srinivasan
Avishay Tal
Madhur Tulsiani
Emanuele Viola
Thomas Watson
Virginia Vassilevska Williams
David Zuckerman

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
COMPUTATIONAL COMPLEXITY CONFERENCE

Average-Case Lower Bounds and Satisfiability Algorithms for Small Threshold Circuits

Ruiwen Chen[*1], **Rahul Santhanam**[*2], **and Srikanth Srinivasan**[3]

1 **University of Oxford, Oxford, UK**
ruiwen.chen@cs.ox.ac.uk
2 **University of Oxford, Oxford, UK**
rahul.santhanam@cs.ox.ac.uk
3 **Indian Institute of Technology Bombay, Mumbai, India**
srikanth@math.iitb.ac.in

Abstract

We show average-case lower bounds for explicit Boolean functions against bounded-depth threshold circuits with a superlinear number of wires. We show that for each integer $d > 1$, there is $\varepsilon_d > 0$ such that Parity has correlation at most $1/n^{\Omega(1)}$ with depth-d threshold circuits which have at most $n^{1+\varepsilon_d}$ wires, and the Generalized Andreev Function has correlation at most $1/2^{n^{\Omega(1)}}$ with depth-d threshold circuits which have at most $n^{1+\varepsilon_d}$ wires. Previously, only worst-case lower bounds in this setting were known [22].

We use our ideas to make progress on several related questions. We give satisfiability algorithms beating brute force search for depth-d threshold circuits with a superlinear number of wires. These are the first such algorithms for depth greater than 2. We also show that Parity cannot be computed by polynomial-size AC^0 circuits with $n^{o(1)}$ *general* threshold gates. Previously no lower bound for Parity in this setting could handle more than $\log(n)$ gates. This result also implies subexponential-time learning algorithms for AC^0 with $n^{o(1)}$ threshold gates under the uniform distribution. In addition, we give almost optimal bounds for the number of gates in a depth-d threshold circuit computing Parity on average, and show average-case lower bounds for threshold formulas of *any* depth.

Our techniques include adaptive random restrictions, anti-concentration and the structural theory of linear threshold functions, and bounded-read Chernoff bounds.

1998 ACM Subject Classification F.1.3 Complexity Measures and Classes

Keywords and phrases threshold circuit, satisfiability algorithm, circuit lower bound

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.1

1 Introduction

One of the main goals in complexity theory is to prove circuit lower bounds for explicit functions in P or NP. We seem quite far from being able to prove that there is a problem in NP that requires superlinear Boolean circuits. We have some understanding, via formulations such as the relativization barrier [5], the "natural proofs" barrier [39] and the algebrization barrier [1], of why current techniques are inadequate for this purpose.

* Work done when the author was at University of Edinburgh, Edinburgh, UK, and supported by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013)/ ERC Grant Agreement no. 615075

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 1; pp. 1:1–1:35

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

However, the community has had more success proving explicit lower bounds against bounded-depth circuits of various kinds. Thanks to pioneering work of Ajtai [2], Furst-Saxe-Sipser [13], Yao [49] and Hastad [19], we know that the Parity and Majority functions require bounded-depth unbounded fan-in circuits of exponential size if only AND and OR gates are allowed. Later Razborov [38] and Smolensky [46] showed that Majority requires exponential size even when MODp gates are allowed in addition to AND and OR gates, for any prime p. The case of bounded-depth circuits with AND, OR and MODm gates, where m is a composite, has been open for nearly thirty years now, even though Majority is conjectured to be hard for such circuits. Williams [47] recently made significant progress by showing that non-deterministic exponential time does not have super-polynomial size circuits with AND, OR and MODm gates, for any m.

For all the bounded-depth circuit classes above, Majority is either known or conjectured to be hard. How about circuit classes which incorporate majority gates, or more generally, gates that are arbitrary linear threshold functions? Note that such gates generalize AND and OR, though not MODp. In the 90s, there was some work on studying the power of bounded-depth threshold circuits. Paturi and Saks [34] showed that depth-2 circuits with majority gates computing Parity require $\tilde{\Omega}(n^2)$ wires; there is also a nearly matching upper bound for Parity. Impagliazzo, Paturi and Saks [22] considered bounded-depth threshold circuits with arbitrary linear threshold gates, and showed that for each depth d, there is a constant $\epsilon_d > 0$ such that Parity requires $n^{1+\epsilon_d}$ wires to compute with depth d threshold circuits.

These lower bounds are *worst case* lower bounds - they show that for any sequence of small circuits, there exist inputs of every length on which the circuits fail to compute Parity. There are several reasons to be interested in *average case* lower bounds under the uniform distribution, or equivalently, in *correlation* upper bounds[1]. For one, average-case lower bounds show that a randomly chosen input is likely to be hard, and thus give a way to generate hard instances efficiently. Second, average-case lower bounds are closely tied to pseudo-random generators via the work of Nisan-Wigderson [30], and are indeed a pre-requisite for obtaining pseudo-random generators with non-trivial seed length for a circuit class. Third, recent work on satisfiability algorithms [42, 20, 7] indicates that the design and analysis of non-trivial satisfiability algorithms is closely tied to proving average-case lower bounds, though there is no formal connection. Fourth, the seminal work of Linial-Mansour-Nisan [26] shows that average-case lower bounds for Parity against a circuit class are tied to non-trivially learning the circuit class under the uniform distribution.

With these different motivations in mind, we systematically study average-case lower bounds for bounded-depth threshold circuits. Our first main result shows correlation upper bounds for Parity and another explicit function known as the Generalized Andreev function with respect to threshold circuits with few wires. No correlation upper bounds for explicit functions against bounded-depth threshold circuits with superlinear wires was known before our work.

▶ **Theorem 1.1.** *For each depth $d \geq 1$, there is a constant $\epsilon_d > 0$ such that for all large enough n, no threshold circuit of depth d with at most $n^{1+\epsilon_d}$ wires agrees with Parity on more than $1/2 + 1/n^{\epsilon_d}$ fraction of inputs of length n, and with the Generalized Andreev function on more than $1/2 + 1/2^{n^{\epsilon_d}}$ fraction of inputs of length n.*

Theorem 1.1 captures the content of Theorem 4.4 and Theorem 4.7 in Section 4.

[1] By contraposition, if any circuit agreeing with a function f on $1/2 + \varepsilon$ of the inputs has size at least s, then size-s circuits have correlation at most 2ε with f, and vice versa.

We constructivize the ideas of the proof of the strong correlation upper bounds for the Generalized Andreev function to get non-trivial satisfiability algorithms for bounded-depth threshold circuits with few wires. Previously, such algorithms were only known for depth 2 circuits, due to Impagliazzo-Paturi-Schneider [21] and Tamaki (unpublished).

▶ **Theorem 1.2.** *For each depth $d \geq 1$, there is a constant $\epsilon_d > 0$ such that the satisfiability of depth-d threshold circuits with at most $n^{1+\epsilon_d}$ wires can be solved in randomized time $2^{n-n^{\epsilon_d}}\mathrm{poly}(n)$.*

Theorem 1.2 is re-stated and proved as Theorem 5.4 in Section 5.

Using our ideas, we also show correlation bounds against AC^0 circuits with a few threshold gates, as well as learning algorithms under the uniform distribution for such circuits.

▶ **Theorem 1.3.** *For each constant d, there is a constant $\gamma > 0$ such that Parity has correlation at most $1/n^{\Omega(1)}$ with AC^0 circuits of depth d and size at most $n^{\log(n)^{0.4}}$ augmented with at most n^γ threshold gates. Moreover, the class of AC^0 circuits of size at most $n^{\log(n)^{0.4}}$ augmented with at most n^γ threshold gates can be learned to constant error under the uniform distribution in time $2^{n^{1/4+o(1)}}$.*

Theorem 1.3 captures the content of Corollary 7.4 and Theorem 7.6 in Section 7.

Having summarized our main results, we now describe related work and our proof techiques in more detail.

1.1 Related work

There has been a large body of work proving upper and lower bounds for constant-depth threshold circuits. Much of this work has focused on the setting of small gate complexity, which seems to be the somewhat easier case to handle. A distinction must also be drawn between work that has focused on the setting where the threshold gates are assumed to be *majority gates* (i.e. the linear function sign representing the gate has integer coefficients that are bounded by a polynomial in the number of variables) and work that focuses on general threshold gates, since analytic tools such as *rational approximation* that are available for majority gates do not work in the setting of general threshold gates.

We discuss the work on wire complexity first, followed by the results on gate complexity.

Wire complexity

Paturi and Saks [34] considered depth-2 *Majority* circuits and showed an $\tilde{\Omega}(n^2)$ lower bound on the wire complexity required to compute Parity; this nearly matches the upper bound of $O(n^2)$. They also showed that there exist majority circuits of size $n^{1+\Theta(\varepsilon_1^d)}$ and depth d computing Parity; here $\varepsilon_1 = 2/(1+\sqrt{5})$. Impagliazzo, Paturi, and Saks [22] showed a depth-d lower bound for *general* threshold circuits computing Parity: namely, that any such circuit must have wire complexity at least $n^{1+\varepsilon_2^d}$ where $\varepsilon_2 < \varepsilon_1$.

The proof of [22] proceeds by induction on the depth d. The main technical lemma shows that a circuit of depth d can be converted to a depth $d-1$ circuit of the same size by setting some of the input variables. The variables that are set are set in a random fashion, but *not* according to the uniform distribution. In fact, this distribution has statistical distance close to 1 from the uniform distribution and furthermore, depends on the circuit whose depth is being reduced. Therefore, it is unclear how to use this technique to prove a correlation bound with respect to the uniform distribution. In contrast, we are able to reduce the depth of the circuit by setting variables uniformly at random (though the variables that we restrict

are sometimes chosen in a way that depends on the circuit), which yields the correlation bounds we want.

Gate complexity

The aforementioned work of Paturi and Saks [34] also proved a near optimal $\tilde{\Omega}(n)$ lower bound on the number of gates in any depth-2 majority circuits computing Parity.

Siu, Roychowdhury, and Kailath [45] considered majority circuits of bounded depth and small gate complexity. They showed that Parity can be computed by depth-d circuits with $O(dn^{1/(d-1)})$ gates. Building on the ideas of [34], they also proved a near matching lower bound of $\tilde{\Omega}(dn^{1/(d-1)})$. Further, they also considered the problem of correlation bounds and showed that there exist depth-d majority circuits with $O(dn^{1/2(d-1)})$ gates that compute Parity almost everywhere and that majority circuits of significantly smaller size have $o(1)$ correlation with Parity (i.e. these circuits cannot compute Parity on more than a $1/2 + o(1)$ fraction of inputs; recall that $1/2$ is trivial since a constant function computes Parity correctly on $1/2$ of its inputs). Impagliazzo, Paturi, and Saks [22] extended the worst case lower bound to general threshold gates, where they proved a slightly weaker lower bound of $\Omega(n^{1/2(d-1)})$. As discussed above, though, it is unclear how to use their technique to prove a correlation bound.

Beigel [6] extended the result of Siu et al. to the setting of AC^0 augmented with a few majority gates. He showed that any subexponential-sized depth-d AC^0 circuit with significantly less than some $k = n^{\Theta(1/d)}$ majority gates has correlation $o(1)$ with Parity. The techniques of all the above works with the exception of [22] were based on the fact majority gates can be well-approximated by low-degree *rational functions*. However, this is not true for general threshold functions [44] and hence, these techniques do not carry over the case of general threshold gates.

A lower bound technique that does carry over to the setting of general threshold gates is that of showing that the circuit class has low-degree polynomial *sign-representations*. Aspnes, Beigel, Furst and Rudich [3] used this idea to prove that AC^0 circuits augmented with a single general threshold *output* gate – we refer to these circuits as TAC^0 circuits as in [15] – of subexponential-size and constant-depth have correlation $o(1)$ with Parity. More recently, Podolskii [36] used this technique along with a trick due to Beigel [6] to prove similar bounds for subexponential-sized AC^0 circuits augmented with general threshold gates. However, this trick incurs an exponential blow-up with the number of threshold gates and hence, in the setting of the Parity function, we cannot handle $k > \log n$ threshold gates.

Another technique that has proved useful in handling general threshold gates is *Communication Complexity*, where the basic idea is to show that the circuit – perhaps after restricting some variables – has low communication complexity in some suitably defined communication model. We can then use results from communication complexity to infer lower bounds or correlation bounds. Nisan [29] used this technique to prove exponential correlation bounds for general threshold circuits (not necessarily even constant-depth) with $n^{1-\Omega(1)}$ threshold gates. Using Beigel's trick and multiparty communication complexity bounds of Babai, Nisan and Szegedy [4], Lovett and Srinivasan [27] (see also [40, 17]) proved exponential correlation bounds for any polynomial-sized AC^0 circuits augmented with up to $n^{\frac{1}{2}-\Omega(1)}$ threshold gates.

We do not use this technique in our setting for many reasons. Firstly, it cannot be used to prove lower bounds or correlation bounds against functions such as Parity (which has small communication complexity in most models). In particular, these ideas do not yield the noise sensitivity bounds we get here. Even more importantly, it is unclear how to use these techniques to prove any sort of superlinear lower bound on wire complexity,

since there are functions that have threshold circuits with linearly many wires, but large communication complexity even after applying restrictions (take a generic read-once depth-2 Majority formula for example).

Perhaps most closely related to our work is that of Gopalan and Servedio [15] who use analytic techniques to prove correlation bounds for AC^0 circuits augmented with a few threshold gates. Their idea is to use Noise sensitivity bounds (as we do as well) to obtain correlation bounds for Parity with TAC^0 circuits and then extend these results in the same way as in the work of Podolskii [36] mentioned above. As a result, though, the result only yields non-trivial bounds when the number of threshold gates is bounded by $\log n$, whereas our result yields correlation bounds for up to $n^{1/2(d-1)}$ threshold gates.

1.2 Proof techniques

In recent years, there has been an explosion of work on the analytic properties (such as Noise Sensitivity) of linear threshold functions (LTFs) and their generalizations polynomial threshold functions (PTFs) (e.g., [43, 33, 9, 18, 10, 28, 23]). We show here that these techniques can be used in the context of constant-depth threshold circuits as well.

Our first result (Theorem 3.1 in Section 3) is a tight correlation bound for Parity with threshold circuits of depth d and gate complexity much smaller than $n^{1/2(d-1)}$. This generalizes both the results of Siu et al. [45], who proved such a result for *majority* circuits, and Impagaliazzo, Paturi, and Saks [22], who proved a worst case lower bound of the same order. The proof uses a fundamental theorem of Peres [35] on the noise sensitivity of LTFs; Peres' theorem has also been used by Klivans, O'Donnell, and Servedio [24] to obtain learning algorithms for functions of a few threshold gates. We use Peres' theorem to prove a noise sensitivity upper bound on small threshold circuits of constant depth.

The observation underlying the proof is that the noise sensitivity of a function is exactly the expected variance of the function after applying a suitable random restriction (see also [31]). Seen in this light, Peres' theorem says that, on application of a random restriction, any threshold function becomes quite biased in expectation and hence is well approximated by a constant function. Our analysis of the threshold circuit therefore proceeds by applying a random restriction to the circuit and replacing all the threshold gates at height 1 by the constants that they are well approximated by to obtain a circuit of depth $d-1$. A straightforward union bound tells us that the new circuit is a good approximation of the original circuit after the restriction. We continue this way with the depth-$d-1$ circuit until the entire circuit becomes a constant, at which point we can say that after a suitable random restriction, the original circuit is well approximated by a constant, which means its variance is small. Hence, the Noise Sensitivity of the original circuit must be small as well and we are done.

This technique is expanded upon in Section 7, where we use a powerful Noise Sensitivity upper bound for low degree PTFs due to Kane [23] along with standard switching arguments [19] to prove similar results for AC^0 circuits augmented with almost $n^{1/2(d-1)}$ threshold gates. This yields Theorem 1.3.

In Section 4, we consider the problem of extending the above correlation bounds to threshold circuits with small (slightly superlinear) wire complexity. The above proof breaks down even for depth-2 threshold circuits with a superlinear number of wires, since such circuits could have a superlinear number of gates and hence the union bound referred to above is no longer feasible.

In the case of depth-2 threshold circuits, we are nevertheless able to use Peres' theorem, along with ideas of [3] to prove correlation bounds for Parity with circuits with nearly $n^{1.5}$

wires. This result is tight, since by the work of Siu et al. [45], Parity can be well approximated by depth-2 circuits with $O(\sqrt{n})$ gates and hence $O(n^{1.5})$ wires. This argument is in Section B.

Unfortunately, however, this technique needs us to set a large number of variables, which renders it unsuitable for larger depths. The reason for this is that, if we set a large number of variables to reduce the depth from some large constant d to $d-1$, then we may be in a setting where the number of wires is much larger than the number of surviving variables and hence correlation bounds with Parity may no longer be possible at all.

We therefore use a different strategy to prove correlation bounds for larger constant depths. The lynchpin in the argument is a qualitative refinement of Peres' theorem (Lemma 4.1) that says that on application of a random restriction to an LTF, with good probability, the variance of the LTF becomes *negligible* (even exponentially small for suitable parameters). The proof of this argument is via anticoncentration results based on the Berry-Esseen theorem and the analysis of general threshold functions via a *critical index* argument as in many recent works [43, 33, 9, 28].

The above refinement of Peres' theorem allows us to proceed with our argument as in the gates case. We apply a random restriction to the circuit and by the refinement, with good probability (say $1 - n^{-\Omega(1)}$) most gates end up exponentially close to constants. We can then set these "imbalanced" gates to constants and still apply a union bound to ensure that the new circuit is a good approximation to the old one. For the small number of gates that do not become imbalanced in this way, we set *all* variables feeding into them. Since the number of such gates is small, we do not set too many variables. We now have a depth $d-1$ circuit. Continuing in this way, we get a correlation bound of $n^{-\Omega(1)}$ with Parity. This gives part of Theorem 1.1.

We then strengthen this correlation bound to $\exp(-n^{\Omega(1)})$ for the *Generalized Andreev function*, which, intuitively speaking, has the following property: even after applying any restriction that leaves a certain number of variables unfixed, the function has exponentially small correlation with any LTF on the surviving variables. To prove lower bounds for larger depth threshold circuits, we follow more or less the same strategy, except that in the above argument, we need most gates to become imbalanced with very high probability $(1 - \exp(-n^{\Omega(1)}))$. To ensure this, we use a *bounded read Chernoff bound* due to Gavinsky, Lovett, Saks, and Srinivasan [14]. We can use this technique to reduce depth as above as long as the number of threshold gates at height 1 is "reasonably large". If the number of gates at height 1 is very small, then we simply guess the values of these few threshold gates and move them to the top of the circuit and proceed. This gives the other part of Theorem 1.1.

This latter depth-reduction lemma can be completely constructivized to design a satisfiability algorithm that runs in time $2^{n-n^{\Omega(1)}}$. The algorithm proceeds in the same way as the above argument, iteratively reducing the depth of the circuit. A subtlety arises when we replace imbalanced gates by constants, since we are changing the behaviour of the circuit on some (though very few) inputs. Thus, a circuit which was satisfiable only at one among these inputs might now end up unsatisfiable. However, we show that there is an efficient algorithm that enumerates these inputs and can hence check if there are satisfiable assignments to the circuits from among these inputs. This gives Theorem 1.2.

In Section 6, we prove correlation bounds for the Generalized Andreev function with threshold *formulas* of any arity and any depth. The proof is based on a retooling of the argument of Nečiporuk for formulas of constant arity over any basis and yields a correlation bound as long as the wire complexity is at most $n^{1.5-\Omega(1)}$.

2 Preliminaries

2.1 Basic Boolean function definitions

A Boolean function on n variables will be a function $f : \{-1,1\}^n \to \{-1,1\}$. We use the standard inner product on functions $f, g : \{-1,1\}^n \to \mathbb{R}$ defined by $\langle f, g\rangle = \mathbf{E}_{x\sim\{-1,1\}^n}[f(x)g(x)]$[2].

Given Boolean functions f, g on n variables, the *Correlation* between f and g – denoted $\mathrm{Corr}(f,g)$ – is defined as

$$\mathrm{Corr}(f,g) := |\langle f, g\rangle| = \left|\mathop{\mathbf{E}}_{x\sim\{-1,1\}^n}[f(x)g(x)]\right| = |2\Pr_x[f(x) = g(x)] - 1|\,.$$

Also, we use $\delta(f,g)$ to denote the fractional distance between f and g: i.e., $\delta(f,g) = \Pr_x[f(x) \neq g(x)]$. Then, we have $\mathrm{Corr}(f,g) = |1 - 2\delta(f,g)|$. We say that f is δ-approximated by g if $\delta(f,g) \leq \delta$.

We use Par_n to denote the parity function on n variables. I.e. $\mathrm{Par}_n(x_1,\ldots,x_n) = \prod_{i=1}^n x_i$.

▶ **Definition 2.1** (Restrictions). A restriction on n variables is a function $\rho : [n] \to \{-1,1,*\}$. A random restriction is a distribution over restrictions. We use $\mathcal{R}_p^n$ to denote the distribution over restrictions on n variables obtained by setting each $\rho(x) = *$ with probability p and to 1 and -1 with probability $\frac{1-p}{2}$ each. We will often view the process of sampling a restriction as picking a pair (I, y) where $I \subseteq [n]$ is obtained by picking each element of $[n]$ to be in I with probability p and $y \in \{-1,1\}^{n-|I|}$ uniformly at random.

▶ **Definition 2.2** (Restriction trees and Decision trees). A *restriction tree* T on $\{-1,1\}^n$ of depth h is a binary tree of depth h all of whose internal nodes are labelled by one of n variables, and the outgoing edges from an internal node are labelled +1 and -1; we assume that a node and its ancestor never query the same variable. Each leaf ℓ of T defines a restriction ρ_ℓ that sets all the variables on the path from the root of the decision tree to ℓ and leaves the remaining variables unset. A random restriction tree $\mathcal{T}$ of depth h is a distribution over restriction trees of depth h.

Given a restriction tree T, the process of choosing a random edge out of each internal node generates a distribution over the leaves of the tree (note that this distribution is not uniform: the weight it puts on leaf ℓ at depth d is 2^{-d}). We use the notation $\ell \sim T$ to denote a leaf ℓ of T picked according this distribution.

A *decision tree* is a restriction tree all of whose leaves are labelled either by +1 or -1. We say a decision tree has size s if the tree has s leaves. We say a decision tree computes a function $f : \{-1,1\}^n \to \{-1,1\}$ if for each leaf ℓ of the tree, $f|_{\rho_\ell}$ is equal to the label of ℓ.

▶ **Fact 2.3** (Facts about correlation). *Let $f, g, h : \{-1,1\}^n \to \{-1,1\}$ be arbitrary.*

1. $\mathrm{Corr}(f,g) \in [0,1]$.
2. *If* $\mathrm{Corr}(f,g) \leq \varepsilon$ *and* $\delta(g,h) \leq \delta$, *then* $\mathrm{Corr}(f,h) \leq \varepsilon + 2\delta$.
3. *Let $g_1,\ldots,g_N$ be Boolean functions such that no two of them are simultaneously true and let h denote their OR. Then,* $\mathrm{Corr}(f,h) \leq \sum_{i=1}^N \max\{\mathrm{Corr}(f,1), \mathrm{Corr}(f,g_i)\}$, *where* 1 *denotes the constant* 1 *function.*
4. *Let $\mathcal{T}$ be any random restriction tree. Then* $\mathrm{Corr}(f,g) \leq \mathbf{E}_{T\sim\mathcal{T},\ell\sim T}[\mathrm{Corr}(f|_{\rho_\ell}, g|_{\rho_\ell})]$.

[2] $x \sim \{-1,1\}^n$ stands for that x is uniform in $\{-1,1\}^n$.

▶ **Definition 2.4** (Noise sensitivity and Variance [32]). Given a Boolean function $f : \{-1,1\}^n \to \{-1,1\}$ and a parameter $p \in [0,1]$, we define the *Noise senstivity of f with noise parameter p* – denoted $\mathrm{NS}_p(f)$ – as follows. Pick $x \in \{-1,1\}^n$ uniformly at random and $y \in \{-1,1\}^n$ by negating (i.e. flipping) each bit of x independently with probability p; we define $\mathrm{NS}_p(f) = \Pr_{(x,y)}[f(x) \neq f(y)]$. The variance of f – denoted $\mathrm{Var}(f)$ – is defined to be $2\mathrm{NS}_{1/2}(f)$.

▶ **Proposition 2.5.** *Let $f : \{-1,1\}^n \to \{-1,1\}$ be any Boolean function. Then,*
1. *For $p \leq 1/2$, $\mathrm{NS}_p(f) = \frac{1}{2}\,\mathbf{E}_{\rho \sim \mathcal{R}^n_{2p}}[\mathrm{Var}(f|_\rho)]$.*
2. *If $p \geq \frac{1}{n}$, then $\mathrm{Corr}(f, \mathrm{Par}_n) \leq O(\mathrm{NS}_p(f))$.*

The above fact is folklore, but we couldn't find explicit proofs in the literature. Therefore we present them in the appendix (see Appendix A).

▶ **Fact 2.6.** *Let $f : \{-1,1\}^n \to \{-1,1\}$ be any Boolean function. Let $p = \min\{\Pr_x[f(x) = 1], \Pr_x[f(x) = -1]\}$ where x is chosen uniformly from $\{-1,1\}^n$. Then, $\mathrm{Var}(f) = \Theta(p)$.*

2.2 Threshold functions and circuits

▶ **Definition 2.7** (Threshold functions and gates). A *Threshold gate* is a gate ϕ labelled with a pair (w, θ) where $w \in \mathbb{R}^m$ for some $m \in \mathbb{N}$ and $\theta \in \mathbb{R}$. The gate computes the Boolean function $f_\phi : \{-1,1\}^m \to \{-1,1\}$ defined by $f_\phi(x) = \mathrm{sgn}(\langle w, x\rangle - \theta)$ (we define $\mathrm{sgn}(0) = -1$ for the sake of this definition). The fan-in of the gate ϕ – denoted $\text{fan-in}(\phi)$ – is m. A *Linear Threshold function* (LTF) is a Boolean function that can be represented by a Threshold gate.

▶ **Definition 2.8** (Threshold circuits). A *Threshold circuit C* is a Boolean circuit whose gates are all threshold gates. There are designated output gates, which compute the functions computed by the circuit. Unless explicitly mentioned, however, we assume that our threshold circuits have a unique output gate. The *gate complexity* of C is the number of (non-input) gates in the circuit, while the *wire complexity* is the sum of all the fan-ins of the various gates.

A *Threshold map* from n to m variables is a depth-1 threshold circuit C with n inputs and m outputs. We say that such a map is *read-k* if each input variable is an input to at most k of the threshold gates in C.

The proof of the following can be found for example in [41].

▶ **Lemma 2.9** ([41]). *The number of distinct linear threshold functions on n bits is at most $2^{O(n^2)}$.*

▶ **Definition 2.10** (Restrictions of threshold gates and circuits). Given a threshold gate ϕ of fan-in m labelled by the pair (w, θ) and a restriction ρ on m variables, we use ϕ_ρ to denote the threshold gate over the variables indexed by $\rho^{-1}(*)$ obtained in the natural way by setting variables according to ρ.

We will also need Peres' theorem, which bounds the Noise Sensitivity of threshold functions.

▶ **Theorem 2.11** (Peres' theorem[35, 32]). *Let $f : \{-1,1\}^n \to \{-1,1\}$ be any LTF. Then,*

$$\mathop{\mathbf{E}}_{\rho \sim \mathcal{R}^n_p}[\mathrm{Var}(f|_\rho)] = \mathrm{NS}_{\frac{p}{2}}(f) = O(\sqrt{p}).$$

Using the above for $p = 1/n$ and Proposition 2.5, we obtain

▶ **Corollary 2.12.** *Let $f : \{-1,1\}^n \to \{-1,1\}$ be any threshold function. Then $\mathrm{Corr}(f, \mathrm{Par}_n) \leq O(n^{-1/2})$.*

2.3 Description lengths and Kolmogorov Complexity

▶ **Definition 2.13** (Kolmogorov Complexity). The Kolmogorov complexity of an n-bit Boolean string x is the length of the shortest bit string of the form (M, w) where M is the description of a Turing Machine and w an input to M such that $M(w) = x$. We use $K(x)$ to denote the Kolmogorov complexity of x.

▶ **Fact 2.14.** *For any $\alpha \in (0,1)$, the fraction of n-bit strings x satisfying $K(x) \leq (1-\alpha)n$ is at most $2^{-\alpha n+1}$.*

▶ **Definition 2.15** (Descriptions of circuits). We can also talk about the description lengths of threshold circuits, which we define as follows. By Lemma 2.9, we know that the number of LTFs on n bits is $2^{O(n^2)}$, and hence we can fix some $O(n^2)$-bit description for each such function. The description of a threshold circuit C is a description of the underlying graph theoretic structure of C followed by the descriptions of the threshold functions computed by each of its gates and the input variables labelling its input gates. We use $\sigma(C)$ to denote the length of this description of C.

▶ **Proposition 2.16.** *For any threshold circuit C with wire complexity at most s on at most n variables, $\sigma(C) = O(s^2 + s \log n)$. If $s \geq n$, then the description length is at most $O(s^2)$.*

Proof. Since the wire complexity is at most s, the graph underlying the circuit can be described using $O(s \log s)$ bits (for example, for each wire, we can describe the gates that it connects). Let $\phi_1, \ldots, \phi_m$ be the threshold gates in the circuit. We can write down a description of the LTFs $f_1, \ldots, f_m$ using $\sum_i O(k_i^2)$ bits where k_i is the fan-in of ϕ_i; this is at most $O(\sum_i k_i)^2 = O(s^2)$. Finally, to describe the input variable assignments to the input gates, we need $O(s \log n)$ bits. ◀

2.4 The Generalized Andreev function

We state here the definition of a generalization of Andreev's function, due to Komargodski and Raz, and Chen, Kabanets, Kolokolova, Shaltiel, and Zuckerman [25, 7]. This function will be used to give strong correlation bounds for constant-depth threshold circuits with slightly superlinear wire complexity.

We first need some definitions.

▶ **Definition 2.17** (Bit-fixing extractor). A function $E : \{-1,1\}^n \to \{-1,1\}^m$ is a (n, k, m, ζ) *bit-fixing extractor* if for every random variable X that is uniform on a subcube[3] of $\{-1,1\}^n$ of dimension at least k, the function $E(X)$ is ζ-close to uniform on $\{-1,1\}^m$.

We have the following explicit construction of a bit-fixing extractor.

▶ **Theorem 2.18** ([37]). *There is an absolute constant $c \geq 1$ so that the following holds. There is a polynomial-time computable function $E : \{-1,1\}^n \to \{-1,1\}^m$ that is an (n, k, m, ζ)-bit fixing extractor for any $k \geq (\log n)^c$, $m = 0.9k$, and $\zeta \leq 2^{-k^{\Omega(1)}}$.*

Also recall that a function $\mathrm{Enc} : \{-1,1\}^a \to \{-1,1\}^b$ defines (α, L)-*error-correcting code* for parameters $\alpha \in [0,1]$ and $L \in \mathbb{N}$ if for any $z \in \{-1,1\}^b$, the number of elements in the image of Enc that are at relative Hamming distance at most α from z is bounded by L.

The following theorem is a folklore result, and stated explicitly in the work of Chen et al. [7].

[3] A subcube of dimension k is a subset of $\{-1,1\}^n$ containing elements which are consistent with some restriction with k $*$'s.

▶ **Theorem 2.19** ([7], Theorem 6.4). *Let $r = n^\beta$ for any fixed $0 < \beta < 1$. There exists an (α, L)-error correcting code with* $\mathrm{Enc} : \{-1,1\}^{4n} \to \{-1,1\}^{2^r}$ *where $\alpha = \frac{1}{2} - O(2^{-r/4})$ and $L = O(2^{r/2})$. Further, there is a* $\mathrm{poly}(n)$ *time algorithm, which when given as input $x \in \{-1,1\}^n$ and $i \in [2^r]$ in binary, outputs* $\mathrm{Enc}(x)_i$*, the ith bit of* $\mathrm{Enc}(x)$.

Now we can define the generalized Andreev function as in [7]. The function is $F : \{-1,1\}^{4n} \times \{-1,1\}^n \to \{-1,1\}$ and is defined as follows. Let $\gamma > 0$ be a constant parameter. The parameter will be fixed later according to the application at hand.

Let E be *any* $(n, n^\gamma, m = 0.9n^\gamma, 2^{-n^{\Omega(\gamma)}})$ extractor (we can obtain an explicit one using Theorem 2.18). We interpret the output of E as an integer from $[2^m]$ in the natural way. Let $\mathrm{Enc} : \{-1,1\}^{4n} \to \{-1,1\}^{2^m}$ define a $(\frac{1}{2} - O(2^{-m/4}), 2^{m/2})$-list decodable code as in Theorem 2.19. Then, we define $F(x_1, x_2)$ by

$$F(x_1, x_2) = \mathrm{Enc}(x_1)_{E(x_2)}. \tag{1}$$

Given $a \in \{-1,1\}^{4n}$, we use $F_a(\cdot)$ to denote the resulting sub-function on n bits obtained by fixing $x_1 = a$.

The following lemma was proved as part of Theorem 6.5 in [7].

▶ **Lemma 2.20** ([7], Theorem 6.5). *Let C be any circuit on n^γ variables with binary description length $\sigma(C) \leq n$ according to some fixed encoding scheme. Let ρ be any restriction of n variables leaving n^γ variables unfixed. Let $f(y) := F_a|_\rho(y)$ for $a \in \{-1,1\}^{4n}$ satisfying $K(a) \geq 3n$. Then*

$$\mathrm{Corr}(f, C) \leq \exp(-n^{\Omega(\gamma)}).$$

2.5 Concentration bounds

We state a collection of concentration bounds that we will need in our proofs. The proofs of Theorems 2.21 and 2.23 may be found in the excellent book by Dubhashi and Panconesi [11].

▶ **Theorem 2.21** (Chernoff bound). *Let $w \in \mathbb{R}^n$ be arbitrary and x is chosen uniformly from $\{-1,1\}^n$. Then*

$$\Pr_x[|\langle w, x\rangle| \geq t \cdot \|w\|_2] \leq \exp(-\Omega(t^2)).$$

▶ **Definition 2.22** (Imbalance). We say that a threshold gate ϕ labelled by (w, θ) is t-imbalanced if $|\theta| \geq t \cdot \|w\|_2$ and t-balanced otherwise.

We also need a multiplicative form of the Chernoff bound for sums of Boolean random variables.

▶ **Theorem 2.23** (Multiplicative Chernoff bound). *Let $Y_1, \ldots, Y_m$ be independent Boolean random variables such that $\mathbf{E}[Y_i] = p_i$ for each $i \in [m]$. Let p denote the average of the p_i. Then, for any $\varepsilon > 0$*

$$\Pr[|\sum_i Y_i - pm| \geq \varepsilon pm] \leq \exp(-\Omega(\varepsilon^2 pm)).$$

Let $Y_1, \ldots, Y_m$ be random variables defined as functions of independent random variables $X_1, \ldots, X_n$. For $i \in [m]$, let $S_i \subseteq [n]$ index those random variables among $X_1, \ldots, X_n$ that influence Y_i. We say that $Y_1, \ldots, Y_m$ are read-k random variables if any $j \in [n]$ belongs to S_i for at most k different $i \in [m]$.

The notation $D(p||q)$ represents the KL-divergence (see, e.g., [8]) between the two probability distributions on $\{0,1\}$ where the probabilities assigned to 1 are p and q respectively.

▶ **Theorem 2.24** (A read-k Chernoff bound [14]). *Let $Y_1, \ldots, Y_m$ be $\{0,1\}$-valued read-k random variables such that $\mathbf{E}[Y_i] = p_i$. Let p denote the average of $p_1, \ldots, p_m$. Then, for any $\varepsilon > 0$,*

$$\Pr[\sum_i Y_i \geq pm(1+\varepsilon)] \leq \exp(-D(p(1+\varepsilon)||p)m/k).$$

Using standard estimates on the KL-divergence, we get

▶ **Corollary 2.25.** *Let $Y_1, \ldots, Y_m$ be as in the statement of Theorem 2.24 and assume $\mathbf{E}[\sum_i Y_i] \leq \mu$. Then,*

$$\Pr[\sum_i Y_i \geq 2\mu] \leq \exp(-\Omega(\mu/k)).$$

3 Correlation bounds for threshold circuits with small gate complexity

In this section, we show that constant-depth threshold circuits with a small number of gates cannot correlate well with the Parity function.

It should be noted that Nisan [29] already proved strong correlation bounds for the *Inner Product* function against *any* threshold circuit (not necessarily constant-depth) with a sub-linear (much smaller than $n/\log n$) number of threshold gates. The idea of the proof is to first show that each threshold gate on n variables has a δ-error randomized protocol with complexity $O(\log(n/\delta))$ [29, Theorem 1]. One can use this to show that any threshold circuit as in the theorem can be written as a decision tree of depth n/k querying threshold functions and hence has a $\exp(-\Omega(k))$-error protocol of complexity at most $n/10$. Standard results in communication complexity imply that any such function can have correlation at most $\exp(-\Omega(k))$ with inner product.

However, such techniques cannot be used to obtain lower bounds or correlation bounds for the parity function, since the parity function has low communication complexity (even in the deterministic setting). An even bigger disadvantage to this technique is that it cannot be used to obtain *any* superlinear lower bound on the wire complexity, since threshold circuits with a linear number of wires can easily compute functions with high communication complexity (such as the Disjointness function).

The techniques we use here can be used to give correlation bounds for the parity function; further, these correlation bounds are nearly tight (Theorem 3.4). In fact, we prove something stronger: we upper bound the noise sensitivity of small constant-depth threshold circuits, which additionally implies the existence of non-trivial learning algorithms [24, 15]. Further, our techniques also imply noise sensitivity bounds for AC^0 circuits augmented with a small number of threshold gates.

In this section, we illustrate our technique with the case of threshold circuits with a small number of gates. The generalizations to AC^0 circuits augmented with a small number of threshold gates are obtained in Section 7.

3.1 Correlation bounds via noise sensitivity

▶ **Theorem 3.1.** *Let C be a depth d threshold circuit with at most k threshold gates. Then, for any parameters $p, q \in [0,1]$, we have*

$$\mathrm{NS}_{p^{d-1}q}(C) \leq O(k\sqrt{p} + \sqrt{q}).$$

Proof. We assume that $q \leq \frac{1}{2}$, since otherwise the statement of the theorem is trivial. We will instead prove that for $p_d := 2p^{d-1}q \in [0,1]$ and $\rho_d \sim \mathcal{R}^n_{p_d}$ (n is the number of input variables to C), we have

$$\mathop{\mathbf{E}}_{\rho_d}[\mathrm{Var}(C|_{\rho_d})] \leq O(k\sqrt{p} + \sqrt{q}). \tag{2}$$

This will imply the theorem, since by Proposition 2.5, we have $\mathrm{NS}_{p^{d-1}q}(C) = \frac{1}{2}\mathbf{E}_{\rho_d}[\mathrm{Var}(C|_{\rho_d})]$.

The proof of (2) is by induction on the depth d of the circuit. The base case $d = 1$ is just Peres' theorem (Theorem 2.11).

Now assume that C has depth $d > 1$. Let k_1 be the number of threshold circuits at height 1 in the circuit. We choose a random restriction $\rho \sim \mathcal{R}^n_p$ and consider the circuit $C|_\rho$. It is easy to check that

$$\mathop{\mathbf{E}}_{\rho_d}[\mathrm{Var}(C|_{\rho_d})] = \mathop{\mathbf{E}}_{\rho}[\mathop{\mathbf{E}}_{\rho_{d-1}}[\mathrm{Var}((C|_\rho)|_{\rho_{d-1}})]], \tag{3}$$

and hence to prove (2), it suffices to bound the expectation of $\mathrm{Var}((C|_\rho)|_{\rho_{d-1}})$.

Let us first consider the circuit $C|_\rho$. Peres' theorem tells us that on application of the restriction ρ, each threshold gate at height 1 becomes quite biased on average. Formally, by Theorem 2.11 and Fact 2.6, for each threshold gate ϕ at height 1, there is a bit $b_{\phi,\rho} \in \{-1,1\}$ such that

$$\mathop{\mathbf{E}}_{\rho}[\Pr_{x \in \{-1,1\}^{|\rho^{-1}(*)|}}[\phi_\rho(x) \neq b_{\phi,\rho}]] \leq O(\sqrt{p}).$$

In particular, replacing ϕ_ρ by $b_{\phi,\rho}$ in the circuit $C|_\rho$ yields a circuit that differs from $C|_\rho$ on only an $O(\sqrt{p})$ fraction of inputs (in expectation). Applying this replacement to each of the k_1 threshold gates at height 1 yields a circuit C'_ρ with $k - k_1$ threshold gates and depth $d-1$ such that

$$\mathop{\mathbf{E}}_{\rho}[\delta(C|_\rho, C'_\rho)] \leq O(k_1\sqrt{p}) \tag{4}$$

where $\delta(C|_\rho, C'_\rho)$ denotes the fraction of inputs on which the two circuits differ. On the other hand, we can apply the inductive hypothesis to C'_ρ to obtain

$$\mathop{\mathbf{E}}_{\rho_{d-1}}[\mathrm{Var}((C'_\rho)|_{\rho_{d-1}})] \leq O((k-k_1)\sqrt{p} + \sqrt{q}). \tag{5}$$

Therefore, to infer (2), we put the above together with (4) and the following elementary fact.

▶ **Proposition 3.2.** *Say $f, g : \{-1,1\}^m \to \{-1,1\}$ and $\delta = \delta(f,g)$. Then, for any $r \in [0,1]$, we have $\mathbf{E}_{\rho\sim\mathcal{R}^n_r}[\mathrm{Var}(f|_\rho)] \leq \mathbf{E}_{\rho\sim\mathcal{R}^n_r}[\mathrm{Var}(g|_\rho)] + 4\delta$.*

Proof of Proposition 3.2. By Proposition 2.5, we know that $\mathbf{E}_{\rho\sim\mathcal{R}^n_r}[\mathrm{Var}(f|_\rho)] = 2\mathrm{NS}_{r/2}(f)$, and similarly for g. By definition of noise sensitivity, we have $\mathrm{NS}_{r/2}(f) = \Pr_{(x,y)}[f(x) \neq f(y)]$ where $x \in \{-1,1\}^m$ is chosen uniformly at random and y is chosen by flipping each bit of x with probability $r/2$. Note that each of x and y is individually uniformly distributed over $\{-1,1\}^m$ and hence, both $f(x) = g(x)$ and $f(y) = g(y)$ hold with probability at least $1 - 2\delta$. This yields

$$\mathrm{NS}_{r/2}(f) = \Pr_{(x,y)}[f(x) \neq f(y)] \leq \Pr_{(x,y)}[g(x) \neq g(y)] + 2\delta = \mathrm{NS}_{r/2}(g) + 2\delta,$$

which implies the claimed bound. ◀◀

▶ **Corollary 3.3.** *Let $d \geq 2$ and $\delta \in [0,1]$ be arbitrary parameters. Assume that C is a depth d threshold circuit over n variables with at most $\delta n^{\frac{1}{2(d-1)}}$ threshold gates. Then,* $\mathrm{Corr}(C, \mathrm{Par}_n) \leq O(\delta^{(1-\frac{1}{d})})$.

Proof. Let $k \leq \delta n^{1/2(d-1)}$ be the number of gates in the threshold circuit C. We apply Theorem 3.1 with the following optimized parameters: $p = \frac{1}{n^{1/d}} \cdot \frac{1}{k^{2/d}}$ and $q \in [0,1]$ such that $p^{d-1}q = \frac{1}{n}$. It may be verified that for this setting of parameters, Theorem 3.1 gives us

$$\mathrm{NS}_{1/n}(C) \leq O\left(\frac{k^{1-1/d}}{n^{1/(2d)}}\right) \leq O(\delta^{1-\frac{1}{d}}).$$

As noted in Proposition 2.5, we have $\mathrm{Corr}(C, \mathrm{Par}_n) \leq O(\mathrm{NS}_{1/n}(C))$. This completes the proof. ◀

▶ Remark. It is instructive to compare the above technique with the closely related work of Gopalan and Servedio [15]. The techniques of [15] applied to the setting of Theorem 3.1 show that $\mathrm{NS}_p(C) \leq O(k2^k\sqrt{p})$, which gives a better dependence on the noise parameter p, but a much worse dependence on k. Indeed, this is not surprising since in this setting, the technique of Gopalan and Servedio does not use the fact that the circuit is of depth d. The threshold circuit is converted to a decision tree of depth k querying threshold functions and it is this tree that is analyzed.

We believe that the right answer should incorporate the best of both bounds: $\mathrm{NS}_p(f) \leq O_d(k^{d-1} \cdot \sqrt{p})$. As in Corollary 3.3, this would show that $\mathrm{Corr}(C, \mathrm{Par}_n) = o(1)$ if $k = o(n^{1/2(d-1)})$, but additionally, we would also get $\mathrm{Corr}(C, \mathrm{Par}_n) \leq n^{-\frac{1}{2}+o(1)}$ as long as $k = n^{o(1)}$, which we are not able to prove currently.

It is known from the work of Siu, Roychowdhury and Kailath [45, Theorem 7] that Corollary 3.3 is tight in the sense that there do exist circuits of gate complexity roughly $n^{1/2(d-1)}$ that have significant correlation with Par_n. More formally,

▶ **Theorem 3.4** (Theorem 7 in [45]). *Let $\varepsilon > 0$ be an arbitrary constant. Then, there is a threshold circuit of depth d with $O(d) \cdot (n \log(1/\varepsilon))^{1/2(d-1)}$ gates that computes Par_n correctly on a $1-\varepsilon$ fraction of inputs.*

4 Correlation bounds for threshold circuits with small wire complexity

The following is a key lemma that will be used in the proofs of our correlation bounds. We state the lemma here and prove our correlation bounds. The lemma will be proved in Section 4.2.

Recall that a threshold gate ϕ with label (w, θ) is t-balanced if $|\theta| \leq t \cdot ||w||_2$.

▶ **Lemma 4.1** (Main Structural lemma for threshold gates). *For any threshold gate ϕ over n variables with label (w, θ) and any $p \in [0,1]$, we have*

$$\Pr_{\rho \sim \mathcal{R}^n_p}[\phi_\rho \text{ is } \frac{1}{p^{\Omega(1)}}\text{-balanced}] \leq p^{\Omega(1)}.$$

The proof of the correlation bounds proceed by iteratively reducing the depth of the circuit. In order to perform this depth-reduction for a depth d circuit, we need to analyze the threshold map defined by the threshold gates at depth $d-1$. The first observation, which follows from Markov's inequality, shows that we may assume (after setting a few variables) that the map reads each variable only a few times.

▶ **Fact 4.2** (Small wire-complexity to small number of reads). *Let C be any threshold circuit on n variables with wire complexity at most cn. Then, there is a set S of at most $n/2$ variables such that each variable outside S is an input variable to at most $2c$ many gates in C.*

The second observation is that if the fan-ins of all the threshold gates are small, then depth-reduction is easy (after setting some more variables).

▶ **Proposition 4.3** (Handling small fan-in gates). *Let $C = (\phi_1, \ldots, \phi_m)$ be any read-k threshold map on n variables such that $\max_i \text{fan-in}(\phi_i) \leq t$. Then, there is a set S of n/kt variables such that each ϕ_i depends on at most one variable in S.*

Proof. This may be done via a simple graph theoretic argument. Define an undirected graph whose vertex set is the set of n variables and two variables are adjacent iff they feed into the same threshold gate. We need to pick an S that is an independent set in this graph. Since the graph has degree at most kt, we can greedily find an independent set of size at least n/kt. Let S be such an independent set. ◀

4.1 Proofs of correlation bounds

Let $B > 2$ be a constant real parameter that we will choose to satisfy various constraints in the proofs below. For $d \geq 1$, define $\varepsilon_d = B^{-(2d-1)}$ and $\delta_d = B\varepsilon_d$.

▶ **Theorem 4.4** (Correlation bounds for parity). *For any $d \geq 1$ and $c \leq n^{\varepsilon_d}$, any depth-d threshold circuit C with at most cn wires satisfies $\text{Corr}(C, \text{Par}_n) \leq O(n^{-\varepsilon_d})$ where the $O(\cdot)$ hides absolute constants (independent of d and n).*

Proof. The proof is by induction on the depth d of C. The base case is $d = 1$, which is the case when C is only a single threshold gate. In this case, Corollary 2.12 tells us that $\text{Corr}(C, \text{Par}_n) \leq O(n^{-1/2}) \leq n^{-\varepsilon_1}$, since $B > 2$.

Now, we handle the inductive case when the depth $d > 1$. Our analysis proceeds in phases.

Phase 1

We first transform the circuit into a read-$2c$ circuit by setting $n/2$ variables. This may be done by Fact 4.2. This defines a restriction tree of depth $n/2$. By Fact 2.3, it suffices to show that each leaf of this restriction tree, the correlation of the restricted circuit and $\text{Par}_{n/2}$ remains bounded by $O(n^{-\varepsilon_d})$.

Let n_1 now denote the new number of variables and let C_1 now be the restricted circuit at some arbitrary leaf of the restriction tree. By renaming the variables, we assume that they are indexed by the set $[n_1]$.

Phase 2

Let $\phi_1, \ldots, \phi_m$ be the threshold gates at depth $d-1$ in the circuit C_1. We call ϕ_i *large* if $\text{fan-in}(\phi_i) > n^{\delta_d}$ and *small* otherwise. Let $L \subseteq [m]$ be defined by $L = \{i \in [m] \mid \phi_i \text{ large}\}$. Assume that $|L| = \ell$. Note that $\ell \cdot n^{\delta_d} \leq n^{1+\varepsilon_d}$ and hence $\ell \leq n^{1+\varepsilon_d-\delta_d} \leq n$.

We restrict the circuit with a random restriction $\rho = (I, y) \sim \mathcal{R}_p^{n_1}$, where $p = n^{-\delta_d/2}$. By Lemma 4.1, we know that for each $i \in [m]$ and some $t = \frac{1}{p^{\Omega(1)}}$ and $q = p^{\Omega(1)}$,

$$\Pr_{\rho}[\phi_i|_\rho \ t\text{-balanced}] \leq q. \tag{6}$$

Further, we also know that for each $i \in L$, the expected value of fan-in$(\phi_i|_\rho) = p \cdot$fan-in(ϕ_i), since each variable is set to a constant with probability $1-p$. Since $i \in L$, the expected fan-in of each ϕ_i ($i \in L$) is at least $n^{\delta_d/2}$. Hence, by a Chernoff bound (Theorem 2.23), we see that for any $i \in L$,

$$\Pr_\rho[\text{fan-in}(\phi_i|_\rho) > 2p \cdot \text{fan-in}(\phi_i)] \leq \exp(-\Omega(n^{\delta_d/2})). \tag{7}$$

Finally another Chernoff bound (Theorem 2.23) tells us that

$$\Pr_{\rho=(I,y)}[|I| < \frac{n_1 p}{2}] \leq \exp(-\Omega(n_1 p)) = \exp(-\Omega(np)). \tag{8}$$

We call a set I *generic* if $|I| \geq \frac{n_1 p}{2}$ and fan-in$(\phi_i|_\rho) \leq 2p \cdot$fan-in$(\phi_i)$ for each $i \in L$. Let $\mathcal{G}$ denote the event that I is generic. By (7) and (8), we know that $\Pr_I[\neg\mathcal{G}] \leq \ell \exp(-\Omega(n^{\delta_d/2})) + \exp(-\Omega(np)) \leq \exp(-n^{\delta_d/4})$. In particular, conditioning on $\mathcal{G}$ doesn't change (6) by much.

$$\Pr_{\rho=(I,y)}[\phi_i|_\rho \text{ } t\text{-balanced} \mid \mathcal{G}] \leq q + \exp(-n^{\delta_d/4}) \leq 2q. \tag{9}$$

Our aim is to further restrict the circuit by setting *all* the input variables to the gates ϕ_i that are t-balanced. In order to analyze this procedure, we define random variables Y_i ($i \in L$) so that $Y_i = 0$ if $\phi_i|_\rho$ is t-imbalanced and fan-in$(\phi_i|_\rho)$ otherwise. Let $Y = \sum_{i \in L} Y_i$. Note that

$$\mathop{\mathbf{E}}_\rho[Y_i \mid \mathcal{G}] \leq (2p \cdot \text{fan-in}(\phi_i)) \cdot \Pr_\rho[\phi_i|_\rho \text{ } t\text{-balanced} \mid \mathcal{G}] \leq 4pq \cdot \text{fan-in}(\phi_i)$$

where the first inequality follows from the fact that since we have conditioned on I being generic, we have fan-in$(\phi_i|_\rho) \leq 2p \cdot$fan-in$(\phi_i)$ with probability 1. Hence, we have

$$\mathop{\mathbf{E}}_\rho[Y \mid \mathcal{G}] \leq 4pq \cdot \sum_i \text{fan-in}(\phi_i) \leq 4pq \cdot n^{1+\varepsilon_d}. \tag{10}$$

We let $\mu := 4pq \cdot n^{1+\varepsilon_d}$. By Markov's inequality,

$$\Pr_\rho[Y \geq \frac{\mu}{\sqrt{q}} \mid \mathcal{G}] \leq \sqrt{q}. \tag{11}$$

In particular, we can condition on a *fixed* generic $I \subseteq [n]$ such that for random $y \sim \{-1,1\}^{n_1 - |I|}$, we have

$$\Pr_y[Y \geq \frac{\mu}{\sqrt{q}}] \leq \sqrt{q}.$$

The above gives us a restriction tree T (that simply sets all the variables in $[n_1] \setminus I$) such that at all but $1 - 2\sqrt{q}$ fraction of leaves λ of T, the total fan-in of the large gates at depth 1 in C_1 that are t-balanced is at most $\frac{\mu}{\sqrt{q}}$; call such λ *good* leaves. Let n_2 denote $|I|$, which is the number of surviving variables.

Phase 3

We will show that for any good leaf λ, we have

$$\text{Corr}(C_\lambda, \text{Par}_{n_2}) \leq n^{-\varepsilon_d} \tag{12}$$

where C_λ denotes $C_1|_{\rho_\lambda}$. This will prove the theorem, since we have by Fact 2.3,

$$\begin{aligned}\mathrm{Corr}(C_1, \mathrm{Par}_{n_1}) &\leq \mathop{\mathbf{E}}_{\lambda\sim T}[\mathrm{Corr}(C_\lambda, \mathrm{Par}_{n_2})] \\ &\leq \Pr_{\lambda}[\lambda \text{ not good}] + \max_{\lambda \text{ good}} \mathrm{Corr}(C_\lambda, \mathrm{Par}_{n_2}) \\ &\leq 2\sqrt{q} + n^{-\varepsilon_d} \leq 2n^{-\varepsilon_d}\end{aligned}$$

where we have used the fact that $\mathrm{Par}_{n_1}|_{\rho_\lambda} = \pm\mathrm{Par}_{n_2}$ for each leaf λ, and also that $2\sqrt{q} \leq n^{-\varepsilon_d}$ for a large enough choice of the constant B.

It remains to prove (12). We do this in two steps.

In the first step, we set all large t-imbalanced gates to their most probable constant values. Formally, for a t-imbalanced threshold gate ϕ labelled by (w, θ), we have $|\theta| \geq t \cdot \|w\|_2$. We replace ϕ by a constant b_ϕ which is 1 if $\theta \geq t \cdot \|w\|_2$ and by -1 if $-\theta \geq t \cdot \|w\|_2$. This turns the circuit C_λ into a circuit C'_λ of at most the wire complexity of C_λ. Further, note that for any $x \in \{-1, 1\}^{n_1}$, $C_\lambda(x) = C'_\lambda(x)$ unless there is a t-imbalanced threshold gate ϕ such that $\phi(x) \neq b_\phi(x)$. By the Chernoff bound (Theorem 2.21) the probability that this happens for any fixed imbalanced threshold gate is at most $\exp(-\Omega(t^2)) \leq \exp(-n^{\Omega(\delta_d)})$. By a union bound over the $\ell \leq n$ large threshold gates, we see that $\Pr_x[C_\lambda(x) \neq C'_\lambda(x)] \leq n\exp(-n^{\Omega(\delta_d)})$. In particular, we get by Fact 2.3

$$\mathrm{Corr}(C_\lambda, \mathrm{Par}_{n_2}) \leq \mathrm{Corr}(C'_\lambda, \mathrm{Par}_{n_2}) + n\exp(-\Omega(n^{\delta_d})) \leq \mathrm{Corr}(C'_\lambda, \mathrm{Par}_{n_2}) + \exp(-n^{\varepsilon_d}). \quad (13)$$

In the second step, we further define a restriction tree T_λ such that C'_λ becomes a depth-$(d-1)$ circuit with at most cn wires at *all the leaves* of T_λ. We first restrict by setting all variables that feed into *any* of the t-balanced gates. The number of variables set in this way is at most

$$\frac{\mu}{\sqrt{q}} \leq 4p\sqrt{q} \cdot n^{1+\varepsilon_d} \leq (pn) \cdot (4\sqrt{q}n^{\varepsilon_d}) \leq \frac{pn}{8} \leq \frac{n_2}{2}$$

for a large enough choice of the constant B. This leaves $n_3 \geq \frac{n_2}{2}$ variables still alive. Further, all the large t-balanced gates are set to constants *with probability* 1. Finally, by Proposition 4.3, we may set all but a set S of $n_4 = n_3/2cn^{\delta_d}$ variables to ensure that with probability 1, all the small gates depend on at most one input variable each. At this point, the circuit C'_λ may be transformed to a depth-$(d-1)$ circuit C''_λ with at most as many wires as C'_λ, which is at most cn.

Note that the number of unset variables is $n_4 \geq pn/8cn^{\delta_d} \geq n^{1-2\delta_d}$, for large enough B. Hence, the number of wires is at most $cn \leq n_4^{\frac{1+\varepsilon_d}{1-2\delta_d}} \leq n_4^{(1+\varepsilon_d)(1+3\delta_d)} \leq n_4^{1+\varepsilon_{d-1}}$ for suitably large B. Thus, by the inductive hypothesis, we have

$$\mathrm{Corr}(C''_\lambda, \mathrm{Par}_{n_4}) \leq O(n_4^{-\varepsilon_{d-1}}) \leq n^{-\varepsilon_d}/2$$

with probability 1 over the choice of the variables restricted in the second step. Along with (13) and Fact 2.3, this implies (12) and hence the theorem. ◀

4.1.1 Strong correlation bounds for the generalized Andreev function

We now prove an exponentially strong correlation bound for the generalized Andreev function defined in Section 2.4 with any $\gamma < 1/6$. As in the case of Theorem 4.4, the proof proceeds by an iterative depth reduction. We prove the depth-reduction in a separate lemma.

▶ **Definition 4.5** (Simplicity)**.** We call a threshold circuit C (t, d, w)-simple if there is a set R of $r \leq t$ threshold functions $g_1, \ldots, g_r$ such that for every setting of these threshold functions to bits $b_1, \ldots, b_r$, the circuit C can be represented on the corresponding inputs (i.e., inputs x satisfying $g_i(x) = b_i$ for each $i \in [r]$) by a depth-d threshold gate of wire complexity at most w.

In particular, note that a (t, d, w)-simple circuit C may be expressed as

$$C(x) = \bigvee_{b_1,\ldots,b_r \in \{-1,1\}} \left(C_{b_1,\ldots,b_r} \wedge \bigwedge_{i:b_i=-1} g_i \wedge \bigwedge_{i:b_i=1} (\neg g_i) \right) \tag{14}$$

where each $C_{b_1,\ldots,b_r}$ is a depth d circuit of wire complexity at most w. Further, note that the OR appearing in the above expression is disjoint (i.e. no two terms in the OR can be simultaneously true).

▶ **Lemma 4.6.** *Let $d \geq 1$ be any constant and assume that ε_d, δ_d are defined as above. Say we are given any depth d threshold circuit C on n variables with at most $n^{1+\varepsilon_d}$ wires.*

There is a restriction tree T of depth $n - n^{1-2\delta_d}$ with the following property: for a random leaf $\lambda \sim T$, let $\mathcal{E}(\lambda)$ denote the event that the circuit $C|_{\rho_\lambda}$ is $\exp(-n^{\varepsilon_d})$-approximated by an $(n^{\delta_d}, d-1, n^{1+\varepsilon_d})$-simple circuit. Then, $\Pr_\lambda[\neg\mathcal{E}(\lambda)] \leq \exp(-n^{\varepsilon_d})$.

Proof. Let $\phi_1, \ldots, \phi_m$ be the threshold gates appearing at height 1 in the circuit C. We say that ϕ_i is large if fan-in$(\phi_i) \geq n^{\delta_d}$ and small otherwise. Let $L = \{i \mid \phi_i \text{ large}\}$ and $S = [m] \setminus L$. Let $\ell = |L|$. Note that $\ell \leq n^{1+\varepsilon_d-\delta_d} \leq n$. Let $c = n^{\varepsilon_d}$.

As in the inductive case of Theorem 4.4, our construction proceeds in phases.

Phase 1

This is identical to Phase 1 in Theorem 4.4. We thus get a restriction tree of depth $n/2$ such that at *all* leaves of this tree, the resulting circuit is a read-$2c$ circuit with at most cn wires. Let C_1 denote the circuit obtained at some arbitrary leaf of the restriction tree and let n_1 denote the number of variables.

Phase 2

This basic idea here is similar to Phase 2 from Theorem 4.4. However, there are technical differences from Theorem 4.4 since we apply a concentration bound to ensure that the circuit simplifies with high probability.

We restrict the circuit with a random restriction $\rho = (I, y) \sim \mathcal{R}_p^{n_1}$, where $p = n^{-\delta_d/2}$. As in Theorem 4.4, we have for some $t = \frac{1}{p^{\Omega(1)}}$, $q = p^{\Omega(1)}$, and for each $i \in [m]$,

$$\Pr_\rho[\phi_i|_\rho \text{ } t\text{-balanced}] \leq q \tag{15}$$

$$\Pr_\rho[\text{fan-in}(\phi_i|_\rho) > 2p \cdot \text{fan-in}(\phi_i)] \leq \exp(-\Omega(n^{\delta_d/2})) \tag{16}$$

$$\Pr_{\rho=(I,y)}[|I| < \frac{n_1 p}{2}] \leq \exp(-\Omega(np)) \tag{17}$$

Now, we partition L as $L = L_1 \cup \cdots \cup L_a$, where $a \leq \frac{1}{\varepsilon_d}$, as follows. The set L_j indexes all threshold gates of fan-in at least $n^{\delta_d+(j-1)\varepsilon_d}$ and less than $n^{\delta_d+j\varepsilon_d}$. We let ℓ_j denote $|L_j|$. For each $i \in L$, let Y_i be a random variable that is 1 if $\phi_i|_\rho$ is t-balanced and 0 otherwise. Note that this defines a collection of read-$2c$ Boolean random variables (the underlying independent random variables are $\rho(k)$ for each $k \in [n_1]$).

Let $Z_j = \sum_{i \in L_j} Y_i$, the number of t-balanced gates in L_j. We have $\mathbf{E}[Z_j] = \sum_{i \in L_j} \mathbf{E}[Y_i] \leq q\ell_j$ by (15). Thus, by an application of the read-$2c$ Chernoff bound in Theorem 2.24, we have

$$\Pr[Z_j \geq 2q\ell_j] \leq \exp\{-\Omega(q\ell_j/c)\}.$$

Assuming that $\ell_j \geq n^{3\delta_d/4}$ and $B = \delta_d/\varepsilon_d$ is a large enough constant, the right hand side of the above inequality is upper bounded bounded by $\exp\{-2n^{\varepsilon_d}\}$. On the other hand if $\ell_j < n^{3\delta_d/4}$, then $Z_j < n^{3\delta_d/4}$ with probability 1. Hence, we have $\Pr_{\rho=(I,y)}[Z_j \geq \max\{2q\ell_j, n^{3\delta_d/4}\}] \leq \exp\{-2n^{\varepsilon_d}\}$ and by a union bound

$$\Pr_{\rho=(I,y)}[\exists j \in [a], Z_j \geq \max\{2q\ell_j, n^{3\delta_d/4}\}] \leq a \exp\{-2n^{\varepsilon_d}\}. \tag{18}$$

We call a set I *generic* if $|I| \geq \frac{n_1 p}{2}$ and fan-in$(\phi_i|_\rho) \leq 2p \cdot$ fan-in(ϕ_i) for each $i \in L$. Let $\mathcal{G}$ denote the event that I is generic. By (16) and (17), we know that $\Pr_I[\neg\mathcal{G}] \leq \ell \exp(-\Omega(n^{\delta_d/2})) + \exp(-\Omega(np)) \leq \exp(-n^{\delta_d/4})$. In particular, similar to Theorem 4.4, we get,

$$\Pr_{\rho=(I,y)}[\exists j \in [a], Z_j \geq \max\{2q\ell_j, n^{3\delta_d/4}\} \mid \mathcal{G}] \leq a \exp\{-2n^{\varepsilon_d}\} + \exp(-n^{\delta_d/4}) \leq 2a \exp\{-2n^{\varepsilon_d}\}. \tag{19}$$

We fix any generic I such that

$$\Pr_y[\exists j \in [a], Z_j \geq \max\{2q\ell_j, n^{3\delta_d/4}\}] \leq 2a \exp\{-2n^{\varepsilon_d}\}. \tag{20}$$

Consider the restriction tree T that sets all the variables not in I. The tree leaves $n_2 \geq pn_1/2 = pn/4$ variables unfixed. We call a leaf λ of the tree *good* if for each $j \in [a]$ we have $Z_j < \max\{2q\ell_j, n^{3\delta_d/4}\}$ and *bad* otherwise. We have

$$\Pr_{\lambda \sim T}[\lambda \text{ a bad leaf}] \leq 2a \exp(-2n^{\varepsilon_d}) \tag{21}$$

For good leaves λ, we show how to approximate $C_\lambda := C_1|_{\rho_\lambda}$ as claimed in the lemma statement.

For the remainder of the argument, fix any good leaf λ. We partition $[a] = J_1 \cup J_2$ where $J_1 = \{j \in [a] \mid Z_j < 2q\ell_j\}$. Note that for any $j \in J_1$, we have

$$\begin{aligned}
\sum_{i \in L_j} Y_i \cdot \text{fan-in}(\phi_i|_{\rho_\lambda}) &\leq \sum_{i \in L_j} Y_i \cdot 2p \cdot \text{fan-in}(\phi_i) \\
&\leq 2p \cdot n^{\delta_d + j\cdot\varepsilon_d} \cdot Z_j \leq n^{\delta_d + j\cdot\varepsilon_d} \cdot 4pq\ell_j \\
&= 4pqn^{\varepsilon_d} \cdot \ell_j \cdot n^{\delta_d + (j-1)\cdot\varepsilon_d} \leq 4pqn^{\varepsilon_d} n^{1+\varepsilon_d} \\
&= 4pqn^{1+2\varepsilon_d}
\end{aligned}$$

where for the last inequality, we have used the fact that since we have ℓ_j gates of fan-in at least $n^{\delta_d+(j-1)\varepsilon_d}$ each, we must have $\ell_j \cdot n^{\delta_d+(j-1)\varepsilon_d} \leq n^{1+\varepsilon_d}$, the total wire complexity of the circuit.

In particular, we can bound the total fan-in of all the t-balanced gates indexed by $\bigcup_{j \in J_1} L_j$ by

$$\sum_{j \in J_1} \sum_{i \in L_j} Y_i \cdot \text{fan-in}(\phi_i|_{\rho_\lambda}) \leq \frac{4pqn^{1+2\varepsilon_d}}{\varepsilon_d}. \tag{22}$$

Phase 3

We proceed in two steps as in Theorem 4.4. Since the steps are very similar, we just sketch the arguments. In the first step, we replace all large t-imbalanced gates by their most probable values. This yields a circuit C'_λ of at most the wire complexity of C_λ and such that

$$\Pr_x[C_\lambda(x) \neq C'_\lambda(x)] \leq \ell \exp(-n^{\Omega(\delta_d)}) \leq n \exp(-n^{\Omega(\delta_d)}) \leq \exp(-n^{\varepsilon_d}). \tag{23}$$

In the second step, we construct another restriction tree rooted at λ that simplifies the circuit to the required form. We first restrict by setting all variables that feed into the t-balanced gates that are indexed by $\bigcup_{j\in J_1} L_j$. By (22), the number of variables set is bounded by

$$\frac{4pqn^{1+2\varepsilon_d}}{\varepsilon_d} \leq \frac{4pn \cdot n^{\varepsilon_d - \Omega(\delta_d)}}{\varepsilon_d} \leq \frac{pn}{8} \leq \frac{n_2}{2}$$

for a large enough choice of the constant B. This sets all the t-balanced gates indexed by $\bigcup_{j\in J_1} L_j$ to constants while leaving $n_3 \geq \frac{n_2}{2}$ variables still alive. Finally, by Proposition 4.3, we may set all but a set of $n_4 = n_3/2cn^{\delta_d}$ variables to ensure that with probability 1, all the small gates depend on at most one input variable each. We may replace the small gates by the unique variable they depend on or a constant (if they do not depend on any variable) without increasing the wire complexity of the circuit. Call the circuit thus obtained C''_λ.

At this point, the only threshold gates at height 1 in the circuit C''_λ are the gates indexed by the t-balanced gates in $\bigcup_{j\in J_2} L_j$. But by the definition of J_2, there can be at most $\frac{1}{\varepsilon_d} \cdot n^{3\delta_d/4} \leq n^{\delta_d}$ of them. For every setting of these threshold gates to constants, the circuit becomes a depth-$(d-1)$ circuit of size at most $n^{1+\varepsilon_d}$. Hence, we have a $(n^{\delta_d}, d-1, n^{1+\varepsilon_d})$-simple circuit, as claimed.

Note that the number of variables still surviving is given by $n_4 \geq pn/8cn^{\delta_d} \geq n^{1-2\delta_d}$, for a large enough choice of the parameter B. Hence, the restriction tree constructed satisfies the required depth constraints.

For a random leaf $\nu \sim T$, the probability $\mathcal{E}(\nu)$ does not occur is at most the probability that in Phase 2, the leaf sampled is bad. By (21), this is bounded by $2a \exp(-2n^{\varepsilon_d}) \leq \exp(-n^{\varepsilon_d})$ as claimed. ◀

We now prove the correlation bound for threshold circuits with the generalized Andreev function. For the sake of induction, it helps to prove a statement that is stronger in two ways: firstly, we consider any function $F_a = F(a, \cdot)$ where $a \in \{-1,1\}^{4n}$ has high Kolmogorov complexity and the input to F_a is further restricted by an arbitrary restriction ρ that leaves a certain number of variables alive; secondly, we prove a correlation bound against circuits which are the AND of a small threshold circuit with a small number of threshold gates.

Formally, say that $f : \{-1,1\}^n \to \{-1,1\}$ is (N, d, t, α)-intractable if for any restriction ρ on n variables that leaves $m \geq N$ variables unset, any depth-d threshold circuit C on m variables of wire complexity at most $m^{1+\varepsilon_d}$, and any set S of at most t threshold functions, we have

$$\mathrm{Corr}(f, C \wedge \bigwedge_{g\in S} g) \leq \alpha.$$

The stronger correlation bound is the following.

▶ **Theorem 4.7** (Generalized version of strong correlation). *Fix any constant $d \geq 1$. Let $a \in \{-1,1\}^{4n}$ be any string with $K(a) \geq 3n$. Then, the function F_a is $(n^{1-\varepsilon_d}, d, n^{\varepsilon_d}, \exp(-n^{\varepsilon_d}/2))$-intractable.*

The proof is by induction on d. The properties of F_a are only used to prove the base case of the theorem, which can then be used to prove the induction case using Lemma 4.6. We prove the base case separately below (we assume that the constant $B > 0$ is large enough so that this implies the base case of the theorem stated above).

▶ **Lemma 4.8** (Base case of induction). *Let $a \in \{-1,1\}^{4n}$ be any string with $K(a) \geq 3n$. Then, the function F_a is $(\sqrt{n}, 1, \sqrt{n}, \exp(-n^{\Omega(1)}))$-intractable.*

Proof. Let $\gamma < 1/6$ in the definition of the Generalized Andreev function in Section 2.4. Let τ be any restriction of n variables leaving $m \geq \sqrt{n}$ variables unfixed. Define $f := F_a|_\tau$. Let C be a conjunction of $\sqrt{n}+1$ threshold gates each on m variables. We wish to prove that

$$\mathrm{Corr}(f, C) \leq \exp(-n^{\Omega(\gamma)}).$$

We build a restriction tree T for C of depth $m - n^\gamma$, by restricting all but n^γ arbitrarily chosen variables. For any leaf ℓ of T, the restricted circuit $C_\ell := C|_{\rho_\ell}$ is a conjunction of $\sqrt{n}+1$ threshold gates each on n^γ variables. By Lemma 2.9, each threshold function can be described using $n^{2\gamma}$ bits. Hence, the entire circuit can be described in a standard way using $(\sqrt{n}+1) \cdot O(n^{2\gamma}) < n$ bits. Then, by Lemma 2.20, we have $\mathrm{Corr}(f|_{\rho_\ell}, C_\ell) \leq \exp(-n^{\Omega(\gamma)})$. By Fact 2.3, we then obtain $\mathrm{Corr}(f, C) \leq \exp(n^{-\Omega(\gamma)})$. ◀

Proof of Theorem 4.7. We only need to prove the inductive case. Assume that $d \geq 2$ is given. Fix any restriction ρ that sets all but $m \geq n^{1-\varepsilon_d}$ variables and let $f = F_a|_\rho$. Let C be a depth-d threshold circuit on the surviving variables of wire complexity at most $m^{1+\varepsilon_d}$. Let S be any set of at most n^{ε_d} threshold functions on the m variables. We need to show that $\mathrm{Corr}(f, C \wedge \bigwedge_{g \in S} g) \leq \exp(-n^{\varepsilon_d/2})$.

Apply Lemma 4.6 to circuit C to find a restriction tree T as guaranteed by the statement of the lemma. By Fact 2.3, we have

$$\begin{aligned}\mathrm{Corr}(f, C \wedge \bigwedge_{g \in S} g) &\leq \mathop{\mathbf{E}}_{\ell \sim T}[\mathrm{Corr}(f_\ell, C_\ell \wedge \bigwedge_{g \in S} g_\ell)] \\ &\leq \Pr_\ell[\neg \mathcal{E}(\ell)] + \max_{\ell : \mathcal{E}(\ell) \text{ holds}} \mathrm{Corr}(f_\ell, C_\ell \wedge \bigwedge_{g \in S} g_\ell) \qquad (24)\end{aligned}$$

where f_ℓ denotes $f|_{\rho_\ell}$ and similarly for C_ℓ and g_ℓ, and $\mathcal{E}(\ell)$ is the event defined in the statement of Lemma 4.6.

Fix any leaf ℓ so that $\mathcal{E}(\ell)$ holds. We want to bound $\mathrm{Corr}(f_\ell, C_\ell \wedge \bigwedge_{g \in S} g_\ell)$. By definition of $\mathcal{E}(\ell)$, we know that C_ℓ is $\exp(-m^{\varepsilon_d})$-approximated by a $(m^{\delta_d}, d-1, m^{1+\varepsilon_d})$-simple circuit C'_ℓ. This implies that $C_\ell \wedge \bigwedge_{g \in S} g_\ell$ is $\exp(-m^{\varepsilon_d})$-approximated by $C'_\ell \wedge \bigwedge_{g \in S} g_\ell$. Hence, we have

$$\mathrm{Corr}(f_\ell, C_\ell \wedge \bigwedge_{g \in S} g_\ell) \leq \mathrm{Corr}(f_\ell, C'_\ell \wedge \bigwedge_{g \in S} g_\ell) + \exp(-m^{\varepsilon_d}). \qquad (25)$$

Further, by the definition of simplicity and its consequence (14), we know that there exist $r \leq m^{\delta_d}$ threshold functions $h^\ell_1, \ldots, h^\ell_r$ such that

$$C'_\ell = \bigvee_{b_1, \ldots, b_r \in \{-1,1\}} C_{b_1, \ldots, b_r} \wedge \bigwedge_{i : b_i = -1} h^\ell_i \wedge \bigwedge_{i : b_i = 1} \neg h^\ell_i$$

where each $C_{b_1,\ldots,b_r}$ is a depth $d-1$-threshold circuit of size at most $m^{1+\varepsilon_d}$ and the OR above is disjoint. This further implies that

$$C'_\ell \wedge \bigwedge_{g\in S} g_\ell = \bigvee_{b_1,\ldots,b_r\in\{-1,1\}} \left(C_{b_1,\ldots,b_r} \wedge \bigwedge_{i:b_i=-1} h_i^\ell \wedge \bigwedge_{i:b_i=1} \neg h_i^\ell \wedge \bigwedge_{g\in S} g_\ell \right) \tag{26}$$

and the OR remains disjoint.

Note that we may apply the induction hypothesis to obtain a bound on the correlation with each term in the OR at this point, since the number of surviving variables is at least $m_1 = m^{1-2\delta_d} \geq n^{1-\varepsilon_d-2\delta_d} \geq n^{1-\varepsilon_{d-1}}$ (throughout, we assume that B is a large enough constant for many of the inequalities to hold); and the wire complexity of each depth-$(d-1)$ circuit $C_{b_1,\ldots,b_r}$ is at most $m^{1+\varepsilon_d} \leq m_1^{(1+\varepsilon_d)/(1-2\delta_d)} \leq m_1^{1+\varepsilon_d+3\delta_d} \leq m_1^{1+\varepsilon_{d-1}}$; further, the number of threshold functions in each term is at most $n^{\varepsilon_d} + n^{\delta_d} < m^{\varepsilon_{d-1}}$. Thus, by the inductive hypothesis, we obtain for any $b_1,\ldots,b_r$,

$$\mathrm{Corr}(f, C_{b_1,\ldots,b_r} \wedge \bigwedge_{i:b_i=-1} h_i^\ell \wedge \bigwedge_{i:b_i=1} \neg h_i^\ell \wedge \bigwedge_{g\in S} g_\ell) \leq \exp(-n^{\varepsilon_{d-1}/2}).$$

Using the fact that the OR in (26) is disjoint, from Fact 2.3, we obtain

$$\mathrm{Corr}(f, C'_\ell \wedge \bigwedge_{g\in S} g_\ell) \leq 2^r \cdot \exp(-n^{\varepsilon_{d-1}/2}) \leq 2^{n^{\delta_d}} \cdot \exp(-n^{\varepsilon_{d-1}/2}) \leq \exp(-n^{\varepsilon_d}).$$

Putting the above together with (24) and (25), we obtain

$$\mathrm{Corr}(f, C \wedge \bigwedge_{g\in S} g) \leq \exp(-m^{\varepsilon_d}) + \exp(-n^{\varepsilon_d}) \leq \exp(-n^{\varepsilon_d/2}).$$

which proves the induction case and hence the theorem. ◀

▶ **Corollary 4.9** (Correlation bounds for Andreev's function). *For any $d \geq 1$, any depth-d threshold circuit C of wire complexity at most $n^{1+\varepsilon_d}$ satisfies* $\mathrm{Corr}(C, F) \leq 2\exp(-n^{\varepsilon_d/2})$.

Proof. For a random $a \in \{-1,1\}^{4n}$, we know by Fact 2.14 that $K(a) \geq 3n$ with probability $1 - \exp(-\Omega(n))$. For each such a, by Theorem 4.7, we have $\mathrm{Corr}(C_a, F_a) \leq \exp(-n^{\varepsilon_d/2})$, where C_a is the circuit obtained by substituting $x_1 = a$ in C. Hence, we have

$$\mathrm{Corr}(C, F) \leq \mathop{\mathbf{E}}_a[\mathrm{Corr}(C_a, F_a)] \leq \exp(-\Omega(n)) + \exp(-n^{\varepsilon_d/2}) \leq 2\exp(-n^{\varepsilon_d/2})$$

as claimed. ◀

4.2 Proof of Main Structural Lemma

We need the following definitions and facts that have appeared many times before in the literature on threshold functions (see, e.g., [9]).

Let $\varepsilon \in [0,1]$ be a real parameter. We say that $w \in \mathbb{R}^n$ is ε-regular if for each $i \in [n]$, $|w_i| \leq \varepsilon \cdot \|w\|_2$.

Assume for simplicity that the co-ordinates of the vector w are sorted so that $|w_1| \geq |w_2| \geq \cdots \geq |w_n|$. Let $w_{>i} \in \mathbb{R}^{n-i}$ denote the vector obtained by removing the first i co-ordinates of w. We define the ε-critical index of w be the least $K = K(\varepsilon)$ so that the vector $w_{>K}$ is ε-regular. Note that $K = 0$ if w is already ε-regular and we define $K = n$ if the ε-critical index is not defined.

We say that an n-variable threshold gate ϕ labelled by (w, θ) is ε-regular if w is. Similarly, the ε-critical index of ϕ is defined to be the ε-critical index of w.

Also, we define $L = L(\varepsilon) = \frac{100 \log^2(1/\varepsilon)}{\varepsilon^3}$ for a large constant A that will be made precise later.

The Berry Esseen theorem (see, e.g., [12]) yields the following standard anticoncentration lemma for linear functions. (See [9, Corollary 2.2] for this particular statement.)

▶ **Lemma 4.10** (Anticoncentration for regular linear functions). *Let $w \in \mathbb{R}^n$ be ε-regular and let $J \subseteq \mathbb{R}$ be any interval. Then,*

$$\left| \Pr_{x \in \{-1,1\}^n} [\langle w, x \rangle \in J] - \Phi(J) \right| \leq O(\varepsilon)$$

where $\Phi(\cdot)$ denotes the cdf of the standard Gaussian with mean 0 and variance $\|w\|_2^2$. In particular, if $|J|$ denotes the length of J, then

$$\Pr_x[\langle w, x \rangle \in J] \leq \frac{|J|}{\|w\|_2} + O(\varepsilon).$$

We now proceed with the proof of Lemma 4.1. We start with an easier case of the lemma for regular threshold gates. Throughout, we work with random restrictions sampled from $\mathcal{R}_p^n$ where $p \in [0, 1]$ is the probability from the statement of Lemma 4.1: equivalently, we pick a pair (I, y) where $I \subseteq [n]$ and $y \in \{-1, 1\}^{n-|I|}$. Let $\varepsilon = p^{\frac{1}{8}}$. Let $t = p^{-\frac{1}{16}}$.

Let the threshold gate ϕ be labelled by pair (w, θ), where $w \in \mathbb{R}^n$. We may assume that the variables of the threshold gate have been sorted so that $|w_1| \geq |w_2| \geq \cdots |w_n|$. Note that after applying a restriction ρ, the threshold gate ϕ_ρ is labelled by pair (w', θ'), where w' is the restriction of w to the coordinates in I and

$$\theta' = \theta'(\rho) = \theta - \langle w'', y \rangle. \tag{27}$$

Above, we use w'' to denote the vector w restricted to the indices in $[n] \setminus I$.

For a random restriction $\rho \sim \mathcal{R}_p^n$, define the following "bad" events:

1. $\mathcal{B}(\rho)$: ϕ_ρ is t-balanced: i.e., $\theta' \leq t \cdot \|w'\|_2$. This is the event whose probability we want to upper bound.
2. $\mathcal{B}_1(\rho)$: $\sum_{i \in I} w_i^2 \geq \sqrt{p} \cdot \|w\|_2^2$.
3. $\mathcal{B}_2^k(\rho)$ (k a parameter): One of the first k variables $x_1, \ldots, x_k$ is set to $*$ by ρ.

We have the following simple upper bounds on the probabilities of some of the above bad events:

- Since each variable is set to $*$ with probability p, we have $\mathbf{E}_\rho[\sum_{i \in I} w_i^2] = p \cdot \|w\|_2^2$. By Markov's inequality, we have $\Pr_\rho[\mathcal{B}_1(\rho)] \leq \sqrt{p}$.
- By a union bound, for any k, we have $\Pr_\rho[\mathcal{B}_2^k(\rho)] \leq pk$.

We start with a simpler subcase of the lemma that follows almost directly from Lemma 4.10. We assume throughout that p is a small enough constant, since otherwise the statement of Lemma 4.1 is trivial.

▶ **Lemma 4.11** (The regular case). *Say that w is ε-regular. Then $\Pr_\rho[\mathcal{B}(\rho)] \leq p^{\Omega(1)}$.*

Proof. We bound $\Pr_\rho[\mathcal{B}(\rho)]$ as follows.

$$\begin{aligned} \Pr_\rho[\mathcal{B}(\rho)] &\leq \Pr_\rho[\mathcal{B}_1(\rho)] + \Pr_\rho[\mathcal{B}(\rho) \mid \neg(\mathcal{B}_1(\rho))] \\ &\leq \sqrt{p} + \Pr_\rho[\mathcal{B}(\rho) \mid \neg(\mathcal{B}_1(\rho))]. \end{aligned} \tag{28}$$

Now, note that the event $\neg\mathcal{B}_1(\rho)$ only depends on the choice of $\rho^{-1}(*) = I$. Hence we can condition on an I so that this event occurs; choosing ρ is now equivalent to choosing a random assignment y to the variables in $[n] \setminus I$.

We have $\theta' = \theta - \langle w'', y\rangle$. Using the fact that $\mathcal{B}_1(\rho)$ doesn't occur, we have

- $\|w''\|_2 \geq \|w\|_2\sqrt{1-\sqrt{p}} \geq \|w\|_2/2$. Using the ε-regularity of w, for each $i \notin I$, we have $|w_i| \leq (\varepsilon)\|w\|_2 \leq 2\varepsilon\|w''\|_2$. Thus, w'' is 2ε-regular.
- $\|w'\|_2 \leq p^{1/4}\|w\|_2 \leq 2p^{1/4}\|w''\|_2$,

Using the above, we can see that the probability that

$$\begin{aligned}\Pr_\rho[\mathcal{B}(\rho) \mid \neg\mathcal{B}_1(\rho)] &\leq \Pr_y[|\theta'| \leq t \cdot \|w'\|_2] \leq \Pr_y[|\theta'| \leq 2tp^{1/4} \cdot \|w''\|_2] \\ &\leq \Pr_y[\langle w'', y\rangle \in [\theta - 2tp^{1/4} \cdot \|w''\|_2, \theta + 2tp^{1/4} \cdot \|w''\|_2]] \\ &\leq 4tp^{1/4} + O(\varepsilon) = O(\varepsilon) = p^{\Omega(1)}\end{aligned}$$

where the final inequality uses the anti-concentration bound in Lemma 4.10. Putting the above together with (28), we are done. ◀

Proof of Lemma 4.1. The proof of the lemma is a standard case analysis based on the ε-critical index of the threshold gate ϕ (see [43, 33, 9, 28]).

The first case is when the critical index $K \leq L$. In this case, we bound the probability of $\mathcal{B}(\rho)$ by

$$\begin{aligned}\Pr_\rho[\mathcal{B}(\rho)] &\leq \Pr_\rho[\mathcal{B}_2^K(\rho)] + \Pr_\rho[\mathcal{B}(\rho) \mid \neg\mathcal{B}_2^K(\rho)] \\ &\leq pK + \Pr_\rho[\mathcal{B}(\rho) \mid \neg\mathcal{B}_2^K(\rho)] \leq \sqrt{p} + \Pr_\rho[\mathcal{B}(\rho) \mid \neg\mathcal{B}_2^K(\rho)] \qquad (29)\end{aligned}$$

where the final inequality follows from the fact that $pK \leq pL \leq \sqrt{p}$ by our choice of parameters. The event $\neg\mathcal{B}_2(\rho)$ only depends on the choice of the sub-restriction $\rho|_{[K]}$ and we can condition on $\rho|_{[K]}$ so that this event occurs. From now on, the random choice will be a restriction $\rho' \sim \mathcal{R}_p^{n-K}$ on the remaining variables.

Since the restricted linear function is now ε-regular by the definition of the ε-critical index, we can apply Lemma 4.11 to conclude that $\Pr_{\rho'}[\mathcal{B}(\rho) \mid \neg\mathcal{B}_2^K(\rho)] \leq p^{\Omega(1)}$. Along with (27), this implies the lemma in the case that $K \leq L$.

The second case is when $K > L$. As in previous cases, we first condition on some bad event not occurring. We have

$$\begin{aligned}\Pr_\rho[\mathcal{B}(\rho)] &\leq \Pr_\rho[\mathcal{B}_2^L(\rho)] + \Pr_\rho[\mathcal{B}(\rho) \mid \neg\mathcal{B}_2^L(\rho)] \\ &\leq pL + \Pr_\rho[\mathcal{B}(\rho) \mid \neg\mathcal{B}_2^L(\rho)] \leq \sqrt{p} + \Pr_\rho[\mathcal{B}(\rho) \mid \neg\mathcal{B}_2^L(\rho)]. \qquad (30)\end{aligned}$$

As in Lemma 4.11, we can condition on a fixed I so that $\neg\mathcal{B}_2^L(\rho)$ occurs (i.e. none of the first L variables belong to I). We then use the following claim that is implicit in [9].

▶ **Proposition 4.12.** *Let $L' = \frac{10r\log(1/\varepsilon)}{\varepsilon^2}$ and assume that $K > L'$. Let y be a random assignment to any set of variables including the first $L' = \frac{10r\log(1/\varepsilon)}{\varepsilon}$ variables. Then, the probability over y that the restricted threshold gate is not $(\frac{1}{\varepsilon})$-imbalanced is at most 2^{-r}.*

Applying the above proposition with $L' = L$ and $r = 10\log(1/\varepsilon)$, we have $\Pr_\rho[\mathcal{B}(\rho) \mid \neg\mathcal{B}_2^L(\rho)] \leq \varepsilon^{10}$. Putting this together with (30), we have the claimed upper bound on $\Pr_\rho[\mathcal{B}(\rho)]$ in the case that $K > L$. ◀

We give a proof sketch of Proposition 4.12 in Section C.

5 Satisfiability algorithms beating brute-force search

In this section, we give satisfiability algorithms beating brute force search for bounded-depth threshold circuits with few wires. Until now, such algorithms were only known for threshold circuits of depth 2. We will assume that each threshold gate on m input bits is given as a pair (w, θ), where $w \in \mathbb{Z}^m$ and $\theta \in \mathbb{Z}$, and θ as well as each component of w has bit complexity $\text{poly}(n)$. Note that this assumption is without loss of generality for a threshold function, and that some assumption on representability of threshold functions is necessary in an algorithmic context.

The satisfiability algorithm relies on an algorithmic version of Lemma 4.6, along with a couple of additional ideas. Essentially, we use the algorithmized version of the lemma to reduce the satisfiability of bounded-depth circuits to satisfiability of ANDs of threshold functions, which we can then solve using a recent result of Williams, stated below.

▶ **Theorem 5.1** ([48]). *There is a deterministic algorithm, which given a bounded-depth circuit C on n variables of size $2^{n^{o(1)}}$ with ANDs, ORs and threshold gates, and with the threshold gates appearing only at the bottom layer, decides if C is satisfiable in time $2^{n-n^{\varepsilon'}}\text{poly}(n)$, where $\varepsilon' > 0$ is a constant that depends only on the depth of the circuit.*

We also need the following fact about threshold gates on n input bits: the set of inputs evaluating to 1 (and dually, the set of inputs evaluating to -1) of a linear threshold gate can be enumerated in time proportional to the number of such inputs, modulo a $\text{poly}(n)$ factor.

▶ **Proposition 5.2.** *Let (w, θ) represent a threshold function ϕ on m input bits, where $w \in \mathbb{Z}^m$ and $\theta \in \mathbb{Z}$ are integers of bit complexity $\text{poly}(m)$. Let S be the set of inputs on which ϕ evaluates to 1. Then S can be enumerated in time $|S|\text{poly}(n)$.*

Proof. We will show how to construct a decision tree for ϕ in time $|S|\text{poly}(n)$, where S is the set of inputs on which ϕ takes value 1. Given a decision tree of size at most $|S|\text{poly}(n)$, it is easy to enumerate the set of inputs on which ϕ takes value 1 in time $|S|\text{poly}(n)$ by scanning through leaves labelled 1 and outputting all assignments corresponding to any such leaf.

The decision tree is constructed recursively as follows. Check if ϕ restricted according to the current partial assignment is satisfiable (in the sense that there is a total assignment consistent with the partial assignment for which ϕ evaluates to 1). Note that satisfiability of a linear threshold gate with polynomial bit complexity of the weights can be done trivially in polynomial time. If the satisfiability check fails, make the current node a leaf and label it with -1. If it succeeds, check if the current partial assignment is falsifiable. If this check fails, make the current node a leaf and label it with 1. Otherwise, branch on an arbitrary unassigned variable and recurse.

Clearly, this decision tree can be constructed with polynomial work at each node, and hence in time $N\text{poly}(n)$, where N is the number of leaves of the tree. We show that $N \leq |S|n$. Indeed, we prove inductively that for any internal node v of the tree of height $h \geq 1$, the number of -1 leaves of the tree rooted at v is at most h times the number of 1 leaves, from which the claim follows as the height of the tree $\leq n$.

For the inductive claim, the base case $h = 1$ is clear as any node at height 1 must have one leaf labelled 1 and the other labelled -1. Assume the claim for height h. Consider a node v at height $h + 1$. Either one of its children is a leaf, or not. If one of the children is a leaf, then the other one v' is not and by the induction hypothesis, since it is of height h, has at most h times as many -1 leaves as 1 leaves. The number of -1 leaves of v is at most one plus the number of -1 leaves of v', and hence at most $h + 1$ times the number of 1 leaves. In case

both children of v are internal nodes, then they are both of height at most h, and by the induction hypothesis, both have at most h times as many -1 leaves as 1 leaves, which implies that the same holds for v. ◀

▶ **Definition 5.3.** We use THR to refer to the class of linear threshold functions. We use AND $\circ$ THR to refer to the class of polynomial-size circuits with an AND gate at the top and threshold gates at the bottom layer.

▶ **Theorem 5.4.** *For each integer $d > 0$, there is a constant $\epsilon_d > 0$ such that satisfiability of a depth d threshold circuit with at most $n^{1+\epsilon_d}$ wires on n variables can be solved by a randomized algorithm in time $2^{n-\Omega(n^{\epsilon_d})}\text{poly}(n)$.*

Proof. As the proof follows the proof of Lemma 4.6 closely, we just give a sketch. Call a circuit depth-d AND $\circ$ THR-skew if the top gate is an AND and all but one child of the top gate is a bottom-level threshold gate, with the possibly exceptional child being a depth-$d-1$ threshold circuit with few wires. We follow the depth reduction argument in the lemma to give a recursive algorithm which reduces satisfiability of polynomial-size depth-d AND $\circ$ THR-skew circuits to the satisfiability of polynomial-size depth-$d-1$ AND $\circ$ THR-skew circuits by appropriately restricting variables.

For the base case $d = 1$, we simply appeal to the algorithm given by Theorem 5.1, which solves satisfiability of AND $\circ$ THR circuits of polynomial size in time $2^{n-n^{\varepsilon'}}\text{poly}(n)$ for some constant $\varepsilon' > 0$.

For the inductive case, we simulate the proof of Lemma 4.6, which performs and analyzes a certain kind of adaptive random restriction. Various bad events might happen at Phases 2 and 3 of this random restriction process, however each step of the restriction process as well as the check that a bad event happens can be implemented in polynomial time. Moreover, the probability that a bad event happens is at most $2^{-n^{\epsilon_d}}$. Whenever a bad event happens, we simply do brute force search on the remaining variables of the circuit, but thanks to the exponentially small probability that a bad event happens, with high probability, we only spend time $2^{n-n^{\epsilon_d}}$ on such brute force searches.

In Phase 3 of the restriction process, we replace imbalanced gates by their most probable values. This changes the functionality of the circuit and might lose us satisfying assignments or give us new invalid satisfying assignments. To get around this, for each such imbalanced gate, we use Proposition 5.2 to efficiently enumerate the inputs evaluating to the minority value for each imbalanced gate, and for each such input check whether it satisfies the original circuit. If it does, we just output 'yes'. We also append to the top gate of the skew circuit a child representing the assignment of the imbalanced gate to its majority value – this needs to be done so that we don't end up with "false positives" in the base case of the recursive algorithm. Although each such false positive can be tested, there might be too many of them, and this could destroy all the savings we accrue through the course of the algorithm. The total time spent in enumerating minority values of imbalanced gates is again at most $2^{n-n^{\epsilon_d}}\text{poly}(n)$, with high probability, using the efficient enumeration and the imbalance property.

Finally, there are a few balanced gates – with high probability at most $O(n^{\delta_d})$ of them – for which we need to try all possible values. This could be expensive, but is compensated for by an increased savings for depth $d-1$, just by setting the constant B large enough in the proof of Lemma 4.6. We also need to set B large enough so that the savings given by the application of Williams' algorithm in the base case overwhelms the loss due to branching on balanced threshold gates at depth $d = 2$.

Thus the total running time, once B is chosen appropriately, is $2^{n-\Omega(n^{\epsilon_d})}\text{poly}(n)$, using the fact that $\epsilon_d < \epsilon_{d-1} < \ldots < \epsilon_2$. ◀

6 Threshold formulas

A threshold formula is a threshold circuit such that the fan-out of each gate is at most 1. A formula can be viewed as a tree. Note that a depth-2 threshold circuit can always be converted to a threshold formula without increasing either the wire complexity or the gate complexity (recall that the gate complexity only measures the number of *non-input* gates).

Let $F\colon \{-1,1\}^{4n} \times \{-1,1\}^n \to \{-1,1\}$ be the generalized Andreev function defined in Section 2.4. Recall that F is constructed with $(n, n^\gamma, m = 0.9n^\gamma, 2^{-n^{\Omega(\gamma)}})$ bit fixing extractor $E\colon \{-1,1\}^n \to \{-1,1\}^m$, and $(1/2 - O(2^{-m/4}), 2^{m/2})$ list decodable code $\text{Enc}\colon \{-1,1\}^{4n} \to \{-1,1\}^{2^m}$.

▶ **Theorem 6.1.** *Any threshold formula on n variables with at most $n^{1.5-\gamma}$ wires for has correlation at most $\exp(-n^{\Omega(\gamma)})$ with the generalized Andreev function.*

Proof. Let C be a threshold formula with n inputs and $s = n^{1.5-\gamma}$ wires. Let L be the number of leaves in the formula tree; then $L \le s \le 2L$. We build a restriction tree T for C up to depth $n - pn$ for $p = n^{\gamma/2}/n$, by greedily restricting the most frequent variables appearing in the formula. Since the most frequent variable appears at least L/n times in C, after restricting one variable, the formula tree has at most $L(1 - 1/n)$ leaves left. Continue until pn variables left unrestricted; then the number of remaining leaves is at most $L \cdot \frac{n-1}{n} \cdot \frac{n-2}{n-1} \cdot \cdots \cdot \frac{pn}{pn+1} = pL$. Thus, for any leaf l of T, the restricted formula $C|_{\rho_l}$ (on $pn = n^{\gamma/2}$ variables) has $s(C|_{\rho_l}) \le 2pL \le 2ps$ wires, and by Proposition 2.16, the description length is at most $O(p^2s^2) \le O(n^{1-\gamma}) < n$. Let $a \in \{-1,1\}^{4n}$ be a string with Kolmogorov complexity $K(a) \ge 3n$, and let $F_a(x) := F(a,x)$. Then, by Lemma 2.20, $\text{Corr}(F_a, C) \le \exp(-n^{\Omega(\gamma)})$.

Therefore, for any formula D with $5n$ inputs and $n^{1.5-\gamma}$ wires, $\Pr_x[F(a,x) = D(a,x)] \le 1/2 + \exp(-n^{\Omega(\gamma)})$. Since a random $a \in \{-1,1\}^{4n}$ has $K(a) \ge 3n$ with probability $1 - 2^{-\Omega(n)}$, the correlation of D and F is at most $2^{-\Omega(n)} + \exp(-n^{\Omega(\gamma)}) = \exp(-n^{\Omega(\gamma)})$. ◀

7 Correlation bounds for AC^0 with a few threshold gates

Following Gopalan and Servedio [15], we define $\text{TAC}^0[k]$ to be the class of constant-depth circuits made up of AND and OR gates and at most k arbitrary threshold gates.

We prove upper bounds on the noise sensitivity of small depth-d $\text{TAC}^0[k]$ circuits for k much smaller than $n^{1/2(d-1)}$. The basic idea is the same as in Theorem 3.1, but we also need to use the following powerful result of Kane [23, Corollary 3].

▶ **Definition 7.1** (Polynomial Threshold functions)**.** A Boolean function $f : \{-1,1\}^n \to \{-1,1\}$ is a degree-D *Polynomial Threshold function* if there is a degree-D polynomial $p(x)$ such that $f(x) = \text{sgn}(p(x))$ for all $x \in \{-1,1\}^n$.

▶ **Lemma 7.2** (Kane [23])**.** *Let f be a degree-D PTF. Then, for any $p > 0$,*

$$\text{NS}_p(f) \le \sqrt{p}(\log(1/p))^{O(D \log D)} 2^{O(D^2 \log D)}.$$

The main theorem of the section is the following.

▶ **Theorem 7.3.** *Fix any constant $d \geq 1$. Let C be a depth-d $\mathrm{TAC}^0[k]$ circuit with at most M gates overall. Then, for any $p, q \in [0,1]$ and any $D \geq 1$, we have*

$$\mathrm{NS}_{p^{d-1}q}(C) \leq O(k\alpha(p,D) + \alpha(q,D) + M(10pD)^D)$$

where $\alpha(p,D) := \sqrt{p}(\log(1/p))^{O(D\log D)}2^{O(D^2\log D)}$ and $O(\cdot)$ hides an absolute constant (independent of d).

Proof. This is a standard switching argument (see, e.g., [19]) augmented with the ideas of Theorem 3.1. We assume throughout that $q \leq \frac{1}{2}$ w.l.o.g. since otherwise $\alpha(q,D) \geq q \geq \frac{1}{2}$ and the claim is trivial.

We say that a threshold gate is a *true* threshold gate if it is not an AND or OR gate.

For any parameters $k_1, d_1, t_1, s_1 \in \mathbb{N}$ with $d_1 \geq 2$, we define $\mathrm{TAC}^0[k_1, d_1, t_1, s_1]$ to be the class of constant-depth circuits made up of AND, OR and threshold gates such that:

- The overall depth is at most d_1,
- The total number of gates at *depth at most* $d_1 - 2$ in the circuit is at most s_1,
- All the true threshold gates are at depth at most $d_1 - 2$ and there are at most k_1 of them, and
- The bottom fan-in of the circuit (i.e. the maximum fan-in of a gate at depth $d_1 - 1$) is at most t_1.

Note that the circuit C in the statement of the theorem is in the class $\mathrm{TAC}^0[k, d+1, 1, M]$, since we may replace the input literals with (say) AND gates of fan-in 1 at the expense of increasing the depth by 1 but in the process satisfying all the criteria of the above definition. We prove the following stronger statement: for any p, q, D as in the statement of the theorem, and any C from the class $\mathrm{TAC}^0[k, d, D, M]$ with $d \geq 2$, we have

$$\mathrm{NS}_{p^{d-2}q}(C) = \mathop{\mathbf{E}}_{\rho_d}[\mathrm{Var}(C|_{\rho_d})] \leq O(k\alpha(p,D) + \alpha(q,D) + M(10pD)^D) \tag{31}$$

where $\rho_d \sim \mathcal{R}^n_{p_d}$ and $p_d := 2p^{d-2}q \in [0,1]$. Proving (31) will clearly prove the theorem.

The proof is by induction on d. The base case is $d = 2$. In this case, since there are no true threshold gates at depth $d-1$ by assumption, a true threshold gate can only occur as the output gate of the circuit C. Since AND and OR gates are also threshold gates, we can assume that the output gate is a threshold gate. The bottom fan-in being at most D implies that each gate at depth 1 can be represented exactly as a polynomial of degree at most D and therefore that the function computed by C is a degree-D PTF. Hence, Lemma 7.2 trivially implies the result.

Now assume $d > 2$. Let $\psi_1, \ldots, \psi_s$ denote the AND and OR gates at depth exactly $d-2$ in the circuit and let $\phi_1, \ldots, \phi_m$ denote the true threshold gates. By assumption $m \leq k$ and $s \leq M$. We sample a random restriction $\rho \sim \mathcal{R}^n_p$ and consider the restricted circuit $C|_\rho$.

Håstad's switching lemma [19] tells us that for each $i \in [s]$, we have

$$\Pr_\rho[\text{DT-depth}(\psi_i|_\rho) \geq D] \leq (10pD)^D, \tag{32}$$

and hence by a union bound,

$$\Pr_\rho[\exists i \in [s] : \text{DT-depth}(\psi_i|_\rho) \geq D] \leq s(10pD)^D. \tag{33}$$

Also, as in the base case, we see that each ϕ_j computes a degree-D PTF. Hence, Lemma 7.2 gives us

$$\mathop{\mathbf{E}}_\rho[\sum_{j\in[m]} \mathrm{Var}(\phi_j|_\rho)] \leq m\alpha(p,D). \tag{34}$$

Consider the circuit C'_ρ obtained from $C|_\rho$ as follows: if there is an $i \in [s]$ such that DT-depth$(\psi_i|_\rho) \geq D$, then C'_ρ is defined to be a trivial circuit that always outputs 1; otherwise, C'_ρ is the depth-$d-1$ circuit obtained from $C|_\rho$ as follows:

- We replace each $\phi_j|_\rho$ by a bit $b_{j,\rho} \in \{-1, 1\}$ so that by Fact 2.6, we have

$$\Pr_{x \in \{-1,1\}^{|\rho^{-1}(*)|}}[\phi_j(x) \neq b_{j,\rho}] \leq O(\mathrm{Var}(\phi_j)),$$

- Since each $\psi_i|_\rho$ is a depth-D decision tree, we can write it as a D-DNF or D-CNF or as a *disjoint sum of terms* of size at most D each. For each gate χ at depth at most $d-3$ that takes ψ_i as an input, we do the following:
  - If χ is an OR gate, then we take the D-DNF representing $\psi_i|_\rho$ and feed the terms of the DNF directly into χ, eliminating the output OR gate of the D-DNF.
  - If χ is an AND gate, we do the same as above, except that we use the D-CNF representation of $\psi_i|_\rho$ and eliminate the output AND gate.
  - If χ is a threshold gate, then we write $\psi_i|_\rho$ as a disjoint sum of terms of size at most D each and feed each of the terms directly to χ. The gate χ now has many inputs in the place of $\psi_i|_\rho$, and the weight given to each of these inputs is the same as the weight given to $\psi_i|_\rho$.

  Note that the above operations do not increase the number of gates at depth at most $d-3$ in the circuit.

Note that C'_ρ has depth $d-1$ and bottom fan-in at most D. Further, the number of gates at depth at most $d-3$ in C'_ρ is at most $M-s$. Hence, C'_ρ is a circuit from the class $\mathrm{TAC}^0[k-m, d-1, D, M]$. We can thus apply the induction hypothesis and obtain

$$\mathop{\mathbf{E}}_{\rho_{d-1}}[\mathrm{Var}(C'_\rho|_{\rho_{d-1}})] \leq O((k-m)\alpha(p,D) + \alpha(q,D) + (M-s)(10pD)^D). \tag{35}$$

To obtain (31), we use

$$\begin{aligned}\mathop{\mathbf{E}}_{\rho_d}[\mathrm{Var}(C)] = \mathop{\mathbf{E}}_{\rho_{d-1}}[\mathop{\mathbf{E}}_{\rho}[\mathrm{Var}(C|_\rho)|_{\rho_{d-1}}]] &\leq \mathop{\mathbf{E}}_{\rho_{d-1}}[\mathop{\mathbf{E}}_{\rho}[\mathrm{Var}(C'_\rho)|_{\rho_{d-1}}]] + O\left(\mathop{\mathbf{E}}_{\rho}[\delta(C|_\rho, C'_\rho)]\right) \\ &= \mathop{\mathbf{E}}_{\rho}[\mathop{\mathbf{E}}_{\rho_{d-1}}[\mathrm{Var}(C'_\rho)|_{\rho_{d-1}}]] + O\left(\mathop{\mathbf{E}}_{\rho}[\delta(C|_\rho, C'_\rho)]\right)\end{aligned} \tag{36}$$

where the inequality follows from Proposition 3.2. Inequality (35) allows us to bound the first term on the right hand size.

It remains to analyze the last term on the right hand side of (36). Define a Boolean random variable $Z = Z(\rho)$ which is 1 iff there is an $i \in [s]$ such that ϕ_i is not a depth-D decision tree. Let $\Delta = \Delta(\rho)$ be the random variable defined by $\Delta := Z + \sum_{j \in [m]} \mathrm{Var}(\phi_j|_\rho)$.

It easily follows from the definition of C'_ρ that for any choice of ρ, either $Z = 1$ – in which case we can trivially bound $\delta(C'_\rho, C|_\rho)$ by 1 – or $\delta(C'_\rho, C|_\rho) \leq \sum_j \delta(\phi_j|_\rho, b_{j,\rho}) = \sum_j \Pr_x[\phi_j|_\rho(x) \neq b_{j,\rho}]$. Hence, for any choice of ρ, we get

$$\delta(C'_\rho, C|_\rho) \leq Z + \sum_{j \in [m]} \Pr_{x \in \{-1,1\}^{|\rho^{-1}(*)|}}[\phi_j|_\rho(x) \neq b_{j,\rho}] \leq O(\Delta).$$

Further, by (33) and (34), we have $\mathbf{E}_\rho[\Delta] \leq O(m\alpha(p,D) + s(10pD)^D)$. Putting this together with (35) and (36)[4] gives the claimed bound. This completes the induction. ◀

[4] Of course, we need to be judicious in our choice of constants in the $O(\cdot)$. We leave this matter to the interested reader.

This yields the following correlation bound as in Corollary 3.3.

▶ **Corollary 7.4.** *The following is true for any constant $d \geq 2$. Say C is a depth-d $\mathrm{TAC}^0[k]$ circuit with at most M gates where $k \leq \delta \cdot n^{1/2(d-1)}$ and $M = n^{o(\sqrt{\log n/\log\log n})}$. Then $\mathrm{Corr}(C, \mathrm{Par}_n) \leq n^{o(1)} \cdot \delta^{1-\frac{1}{d}}$. In particular, if $\delta = n^{-\Omega(1)}$, then $\mathrm{Corr}(C, \mathrm{Par}_n) = n^{-\Omega(1)}$.*

Proof. We choose a $D = o(\sqrt{\log n/\log\log n})$ so that $M \leq n^{o(D)}$ and p, q as in Corollary 3.3. We can then use Theorem 7.3 to obtain $\mathrm{NS}_{1/n}(C) \leq n^{o(1)} \cdot \delta^{1-\frac{1}{d}} + M \cdot (10pD)^D$. Since the latter term is $\frac{1}{n^{\omega(1)}}$, we get $\mathrm{NS}_{1/n}(C) \leq n^{o(1)}\delta^{1-\frac{1}{d}}$. By Proposition 2.5, we have $\mathrm{Corr}(C, \mathrm{Par}_n) \leq O(\mathrm{NS}_{1/n}(C))$, which proves the claim. ◀

▶ Remark. The above corollary can be strengthened considerably if a widely believed strengthening of Lemma 7.2 – named the Gotsman-Linial conjecture [16] – is known to hold. The Gotsman-Linial conjecture is a conjecture about the *average sensitivity* of low-degree PTFs. We do not recall the exact statement of the conjecture here, and refer the reader to the work of Gopalan and Servedio [15] instead. As noted by [15, Corollary1 13], the Gotsman-Linial conjecture implies that for any p and any degree D PTF, we have $\mathrm{NS}_p(f) \leq O(D\sqrt{p})$. Plugging in this bound in place of Lemma 45, it is not hard to see that we can obtain $\mathrm{Corr}(C, \mathrm{Par}_n) = o(1)$ for any circuit C of size $2^{n^{o(1)}}$ from the class $\mathrm{TAC}^0[k]$ where $k = n^{1/2(d-1)-\Omega(1)}$. This is almost a complete generalization of the result of Beigel [6] who proved such a result in the setting where all the threshold gates are of polynomial weight. In contrast, the results of Podolskii [36] and Gopalan and Servedio [15] can prove such correlation bounds only if $k < \log n$.

7.1 Learning algorithms for $\mathrm{TAC}^0[k]$ circuits

Theorem 7.3 also allows us to obtain an algorithm to learn small $\mathrm{TAC}^0[k]$ circuits under the uniform distribution via an observation of Klivans, O'Donnell, and Servedio [24]. We have the following lemma that can be obtained by putting together Fact 9 and Corollary 15 in [24].

▶ **Lemma 7.5.** *Let $\mathcal{F}$ be a class of Boolean functions defined on $\{-1,1\}^n$. Assume that we know that for some $\varepsilon > 0$ and $f \in \mathcal{F}$, there is a $\gamma > 0$ such that $\mathrm{NS}_\gamma(f) \leq \varepsilon/3$. Then, there is an algorithm that learns $\mathcal{F}$ with error ε in time $n^{O(1/\gamma)}$.*

Using the above lemma and Theorem 7.3, we get subexponential-time (i.e. $2^{o(n)}$-time) learning algorithms for $\mathrm{TAC}^0[k]$ circuits of small size.

▶ **Theorem 7.6.** *Let d be any fixed constant. The class of $\mathrm{TAC}^0[k]$ circuits of depth d and size M where $M = n^{o(\sqrt{\log n/\log\log n})}$ and $k = \delta n^{1/2(d-1)}$ for some $\delta > 0$ can be learned to within error $\varepsilon > 0$ in time $n^{O(m)}$ where $m = \max\{n^{1+o(1)}\delta^{2(d-1)}/\varepsilon^{2d}, n^{1/4+o(1)}/\varepsilon^2\}$. In particular, if $\delta = n^{-\Omega(1)}$ and $\varepsilon = \Omega(1)$, then the running time of the algorithm is $2^{o(n)}$.*

Proof. We can assume that $\varepsilon \geq 1/n^{1/2d}$ since otherwise, we can just run a brute force algorithm that takes time $2^{O(n)}$. We choose a $D = o(\sqrt{\log n/\log\log n})$ so that $M = n^{o(D)}$. Theorem 7.3 tells us that for any $p, q \geq \frac{1}{n}$ any C from the class of circuits described in the theorem statement, we have

$$\mathrm{NS}_{p^{d-1}q}(C) \leq Ak\sqrt{p} + B\sqrt{q} + O(M(10pD)^D)$$

where A and B are $n^{o(1)}$.

We choose p, q so that the first two terms above are each bounded by $\varepsilon/10$. This requires $p \leq \varepsilon^2/O(k^2 A)$ and $q \leq \varepsilon^2/O(B)$. Further, to ensure that the last term is at most $\varepsilon/10$, it suffices to choose $p \leq n^{-\Omega(1)}$ (in fact, this ensures that the third term is $n^{-\omega(1)}$ whereas $\varepsilon \geq n^{-1/2d}$ by assumption). Thus, we fix $p = \min\{\varepsilon^2/O(k^2 A), n^{-1/4d}\}$ and $q = \varepsilon^2/O(B)$ so that all the above conditions are satisfied. This gives

$$\mathrm{NS}_\gamma(C) \leq \varepsilon/3$$

where $\gamma = p^{d-1}q$. Hence, by Lemma 7.5, we obtain the statement of the theorem. ◀

▶ Remark. Assuming the Gotsman Linial conjecture, the above technique yields subexponential time constant-error learning algorithms as long as $M \leq 2^{n^{o(1)}}$ and $\delta = n^{-\Omega(1)}$. To contrast again with the work of Gopalan and Servedio [15], the results of [15] – even assuming the Gotsman Linial conjecture – only yield subexponential time learning algorithms in the setting where $k < \log n$. However, the dependence on the error parameter in [15] is better than the dependence we obtain here (the running time there has a ε^3 in place of the ε^{2d} that we obtain here).

Acknowledgements. Work of the first and second authors supported by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) / ERC Grant Agreement No. 615075.

References

1 Scott Aaronson and Avi Wigderson. Algebrization: A new barrier in complexity theory. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC'08)*, 2008.

2 Miklos Ajtai. Σ^1_1-formulae on finite structures. *Annals of Pure and Applied Logic*, 24:1–48, 1983.

3 James Aspnes, Richard Beigel, Merrick L. Furst, and Steven Rudich. The expressive power of voting polynomials. *Combinatorica*, 14(2):135–148, 1994. doi:10.1007/BF01215346.

4 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. doi:10.1016/0022-0000(92)90047-M.

5 Theodore Baker, John Gill, and Robert Solovay. Relativizations of the P =? NP question. *SIAM Journal on Computing*, 4(4):431–442, 1975.

6 Richard Beigel. When do extra majority gates help? polylog(N) majority gates are equivalent to one. *Computational Complexity*, 4:314–324, 1994. doi:10.1007/BF01263420.

7 Ruiwen Chen, Valentine Kabanets, Antonina Kolokolova, Ronen Shaltiel, and David Zuckerman. Mining circuit lower bound proofs for meta-algorithms. *Computational Complexity*, 24(2):333–392, 2015. doi:10.1007/s00037-015-0100-0.

8 Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley-Interscience, New York, USA, 2000.

9 Ilias Diakonikolas, Parikshit Gopalan, Ragesh Jaiswal, Rocco A. Servedio, and Emanuele Viola. Bounded independence fools halfspaces. *SIAM J. Comput.*, 39(8):3441–3462, 2010. doi:10.1137/100783030.

10 Ilias Diakonikolas, Prasad Raghavendra, Rocco A. Servedio, and Li-Yang Tan. Average sensitivity and noise sensitivity of polynomial threshold functions. *SIAM J. Comput.*, 43(1):231–253, 2014. doi:10.1137/110855223.

11 Devdatt P. Dubhashi and Alessandro Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms*. Cambridge University Press, 2009. URL: `http://www.cambridge.org/gb/knowledge/isbn/item2327542/`.

12 W. Feller. *An Introduction to Probability Theory and its Applications*. John Wiley & Sons, New York, 3 edition, 1968.

13 Merrick Furst, James Saxe, and Michael Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical Systems Theory*, 17(1):13–27, April 1984.

14 Dmitry Gavinsky, Shachar Lovett, Michael E. Saks, and Srikanth Srinivasan. A tail bound for read-k families of functions. *Random Struct. Algorithms*, 47(1):99–108, 2015. `doi:10.1002/rsa.20532`.

15 Parikshit Gopalan and Rocco A. Servedio. Learning and lower bounds for ac^0 with threshold gates. *Electronic Colloquium on Computational Complexity (ECCC)*, 17:74, 2010. URL: `http://eccc.hpi-web.de/report/2010/074`.

16 Craig Gotsman and Nathan Linial. Spectral properties of threshold functions. *Combinatorica*, 14(1):35–50, 1994. `doi:10.1007/BF01305949`.

17 Kristoffer Arnsfelt Hansen and Peter Bro Miltersen. Some meet-in-the-middle circuit lower bounds. In *Mathematical Foundations of Computer Science 2004, 29th International Symposium, MFCS 2004, Prague, Czech Republic, August 22-27, 2004, Proceedings*, pages 334–345, 2004. `doi:10.1007/978-3-540-28629-5_24`.

18 Prahladh Harsha, Adam Klivans, and Raghu Meka. Bounding the sensitivity of polynomial threshold functions. *Theory of Computing*, 10:1–26, 2014. URL: `http://theoryofcomputing.org/articles/v010a001/`.

19 Johan Håstad. Almost optimal lower bounds for small depth circuits. In *Proceedings of the 18th Annual ACM Symposium on Theory of Computing, May 28-30, 1986, Berkeley, California, USA*, pages 6–20, 1986. `doi:10.1145/12130.12132`.

20 Russell Impagliazzo, William Matthews, and Ramamohan Paturi. A satisfiability algorithm for AC0. In *Proceedings of 23rd Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 961–972, 2012.

21 Russell Impagliazzo, Mohan Paturi, and Stefan Schneider. A satisfiability algorithm for sparse depth two threshold circuits. In *Proceedings of 54th Annual IEEE Symposium on Foundations of Computer Science*, pages 479–488, 2013.

22 Russell Impagliazzo, Ramamohan Paturi, and Michael E. Saks. Size-depth tradeoffs for threshold circuits. *SIAM J. Comput.*, 26(3):693–707, 1997. `doi:10.1137/S0097539792282965`.

23 Daniel M. Kane. The correct exponent for the gotsman-linial conjecture. *Computational Complexity*, 23(2):151–175, 2014. `doi:10.1007/s00037-014-0086-z`.

24 Adam R. Klivans, Ryan O'Donnell, and Rocco A. Servedio. Learning intersections and thresholds of halfspaces. *J. Comput. Syst. Sci.*, 68(4):808–840, 2004. `doi:10.1016/j.jcss.2003.11.002`.

25 Ilan Komargodski and Ran Raz. Average-case lower bounds for formula size. In *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 171–180, 2013. `doi:10.1145/2488608.2488630`.

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

27 Shachar Lovett and Srikanth Srinivasan. Correlation bounds for poly-size AC^0 circuits with $n^{1-o(1)}$ symmetric gates. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques – 14th International Workshop, APPROX 2011, and 15th International Workshop, RANDOM 2011, Princeton, NJ, USA, August 17-19, 2011. Proceedings*, pages 640–651, 2011. `doi:10.1007/978-3-642-22935-0_54`.

28 Raghu Meka and David Zuckerman. Pseudorandom generators for polynomial threshold functions. *SIAM J. Comput.*, 42(3):1275–1301, 2013. doi:10.1137/100811623.

29 Noam Nisan. The communication compelxity of threshold gates. *Combinatorics: Paul Erdös is Eighty, Bolyai Society Mathematical Studies*, pages 301–315, 1993.

30 Noam Nisan and Avi Wigderson. Hardness vs randomness. *Journal of Computer and System Sciences*, 49(2):149–167, 1994.

31 Ryan O'Donnell. Hardness amplification within np. *J. Comput. Syst. Sci.*, 69(1):68–94, 2004. doi:10.1016/j.jcss.2004.01.001.

32 Ryan O'Donnell. *Analysis of Boolean Functions*. Cambridge University Press, 2014. URL: http://www.cambridge.org/de/academic/subjects/computer-science/algorithmics-complexity-computer-algebra-and-computational-g/analysis-boolean-functions.

33 Ryan O'Donnell and Rocco A. Servedio. The chow parameters problem. *SIAM J. Comput.*, 40(1):165–199, 2011. doi:10.1137/090756466.

34 Ramamohan Paturi and Michael E. Saks. Approximating threshold circuits by rational functions. *Inf. Comput.*, 112(2):257–272, 1994. doi:10.1006/inco.1994.1059.

35 Yuval Peres. Noise stability of weighted majority, December 19 2004. Comment: six pages. URL: http://arxiv.org/abs/math/0412377.

36 Vladimir V. Podolskii. Exponential lower bound for bounded depth circuits with few threshold gates. *Inf. Process. Lett.*, 112(7):267–271, 2012. doi:10.1016/j.ipl.2011.12.011.

37 Anup Rao. Extractors for low-weight affine sources. In *Proceedings of the 24th Annual IEEE Conference on Computational Complexity, CCC 2009, Paris, France, 15-18 July 2009*, pages 95–101, 2009. doi:10.1109/CCC.2009.36.

38 Alexander Razborov. Lower bounds on the size of bounded-depth networks over the complete basis with logical addition. *Mathematical Notes of the Academy of Sciences of the USSR*, 41(4):333–338, 1987.

39 Alexander Razborov and Steven Rudich. Natural proofs. *Journal of Computer and System Sciences*, 55(1):24–35, 1997.

40 Alexander A. Razborov and Avi Wigderson. nˆomega(log n) lower bounds on the size of depth-3 threshold circuits with AND gates at the bottom. *Inf. Process. Lett.*, 45(6):303–307, 1993. doi:10.1016/0020-0190(93)90041-7.

41 Michael E. Saks. Slicing the hypercube. *Surveys in Combinatorics, 1993*, pages 211–255, 1993. URL: http://dl.acm.org/citation.cfm?id=164558.164579.

42 Rahul Santhanam. Fighting perebor: New and improved algorithms for formula and QBF satisfiability. In *Proceedings of 51st Annual IEEE Symposium on Foundations of Computer Science*, pages 183–192, 2010.

43 Rocco A. Servedio. Every linear threshold function has a low-weight approximator. *Computational Complexity*, 16(2):180–209, 2007. doi:10.1007/s00037-007-0228-7.

44 Alexander A. Sherstov. Optimal bounds for sign-representing the intersection of two halfspaces by polynomials. *Combinatorica*, 33(1):73–96, 2013. doi:10.1007/s00493-013-2759-7.

45 Kai-Yeung Siu, Vwani P. Roychowdhury, and Thomas Kailath. Rational approximation techniques for analysis of neural networks. *IEEE Transactions on Information Theory*, 40(2):455–466, 1994. doi:10.1109/18.312168.

46 Roman Smolensky. Algebraic methods in the theory of lower bounds for boolean circuit complexity. In *Proceedings of the 19th Annual Symposium on Theory of Computing*, pages 77–82, 1987.

47 Ryan Williams. Non-uniform ACC circuit lower bounds. In *Proceedings of 26th Annual IEEE Conference on Computational Complexity*, pages 115–125, 2011.

48 Ryan Williams. New algorithms and lower bounds for circuits with linear threshold gates. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 – June 03, 2014*, pages 194–202, 2014. doi:10.1145/2591796.2591858.

49 Andrew Yao. Separating the polynomial-time hierarchy by oracles. In *Proceedings of 26th Annual IEEE Symposium on Foundations of Computer Science*, pages 1–10, 1985.

A Proof of Proposition 2.5

Proof. For point 1., we know that $\mathrm{NS}_p(f) = \Pr_{(x,y)}[f(x) \neq f(y)]$ where x and y are sampled as in Definition 2.4. Alternately, we may also think of sampling (x, y) in the following way: choose $\rho = (I, z) \sim \mathcal{R}^n_{2p}$ and for the locations indexed by I we choose $x', y' \in \{-1, 1\}^{|I|}$ independently and uniformly at random to define strings x and y respectively. Hence, we have

$$\mathrm{NS}_p(f) = \Pr_{x,y}[f(x) \neq f(y)] = \mathop{\mathbf{E}}_{\rho}[\Pr_{x',y'}[f|_\rho(x') \neq f|_\rho(y')]] = \mathop{\mathbf{E}}_{\rho}[\frac{1}{2}\mathrm{Var}(f|_\rho)].$$

We now proceed with point 2.. As $\mathrm{NS}_p(f)$ is a decreasing function of p [32], we may assume that $p = \frac{1}{n} \leq \frac{1}{2}$ and hence we have $\mathrm{NS}_{1/n}(f) = \frac{1}{2}\,\mathbf{E}_{\rho\sim\mathcal{R}^n_{2/n}}[\mathrm{Var}(f|_\rho)]$. Note that for $\rho = (I, y)$ chosen as above, the probability that $I \neq \emptyset$ is $\Omega(1)$. Hence we have

$$\mathrm{NS}'_{1/n}(f) := \frac{1}{2} \mathop{\mathbf{E}}_{\rho\sim\mathcal{R}^n_{2/n}}[\mathrm{Var}(f|_\rho) \mid I \neq \emptyset] \leq \frac{\mathrm{NS}_{1/n}(f)}{\Pr_I[I \neq \emptyset]} = O(\mathrm{NS}_{1/n}(f)).$$

Further, note that for any $m \geq 1$ and any Boolean function $g : \{-1, 1\}^m \to \{-1, 1\}$, its distance from either the constant function 1 or the constant function -1 is at most $\mathrm{Var}(g)/2$. Since Par_m has correlation 0 with any constant function, using Fact 2.3, we have $\mathrm{Corr}(\mathrm{Par}_m, g) \leq \mathrm{Var}(g)/2$.

Using Fact 2.3 again, we get

$$\begin{aligned}\mathrm{Corr}(\mathrm{Par}_n, f) &\leq \mathop{\mathbf{E}}_{\rho\sim\mathcal{R}^n_{2/n}}[\mathrm{Corr}(\mathrm{Par}_n|_\rho, f|_\rho) \mid I \neq \emptyset] = \mathop{\mathbf{E}}_{\rho\sim\mathcal{R}^n_{2/n}}[\mathrm{Corr}(\mathrm{Par}_{|I|}, f|_\rho) \mid I \neq \emptyset] \\ &\leq \mathop{\mathbf{E}}_{\rho\sim\mathcal{R}^n_{2/n}}[\frac{1}{2}\mathrm{Var}(g) \mid I \neq \emptyset] = \mathrm{NS}'_{1/n}(f) = O(\mathrm{NS}_{1/n}(f)).\end{aligned}$$

◀

B Correlation bounds for depth-2 threshold circuits

In this section, we prove near optimal correlation bounds for depth-2 threshold circuits computing Parity.

▶ **Theorem B.1** (Main). *Fix any constant $\varepsilon < \frac{1}{2}$. Let $\gamma = \frac{1}{2} - \varepsilon$. Any depth-2 threshold circuit on n variables with at most $n^{1+\varepsilon}$ wires has correlation at most $n^{-\Omega(\gamma)}$ with the parity function on n variables.*

Note that the above theorem is tight, since by Theorem 3.4, there is a depth-2 circuit with $O(\sqrt{n})$ gates (and hence $O(n^{3/2})$ wires) that computes Parity correctly with high probability.

The proof is based on the following two subclaims:

▶ **Theorem B.2** (Aspnes, Beigel, Furst, and Rudich [3]). *Any degree-t polynomial threshold function (PTF) has correlation at most $O(t/\sqrt{m})$ with the parity function on m variables.*

We say that a circuit C is *δ-approximated* by a circuit C' if $\Pr_x[C(x) \neq C'(x)] \leq \delta$.

▶ **Claim B.3.** *Let ε, γ be as in the statement of Theorem B.1 and let α denote $\gamma/3$. Say C denotes a depth-2 threshold circuit of wire complexity $n^{1+\varepsilon}$ and let $f_1, \ldots, f_t$ be the LTFs computed by C at depth-1. Under a random restriction ρ with $*$-probability $p = \frac{1}{n^{1-\alpha}}$, with probability at least $1 - n^{-\Omega(\gamma)}$, the circuit $C|_\rho$ is $n^{-\Omega(\gamma)}$-approximated by a circuit $\tilde{C}_\rho$ which is obtained from C by replacing each of the $f_i|_\rho$s by an $O(n^{\alpha/2-\Omega(\gamma)})$-junta g_i.*

Assuming the above two claims, we can finish the proof of Theorem B.1 easily as follows.

Let C be a circuit of wire complexity $n^{1+\varepsilon}$. We apply a random restriction ρ with $*$-probability $p = \frac{1}{n^{1-\alpha}}$ as in Claim B.3. Call the restriction *good* if there is a circuit $\tilde{C}_\rho$ as in the Claim that $n^{-\Omega(\gamma)}$-approximates $C|_\rho$ and *bad* otherwise. The probability that we have a bad restriction is at most $n^{-\Omega(\gamma)}$.

Say ρ is a good restriction. The circuit $\tilde{C}_\rho$ can be represented by an $O(n^{\alpha/2-\Omega(\gamma)})$-degree PTF and hence by Theorem B.2 has correlation at most $n^{-\Omega(\gamma)}$ with parity (on the remaining n^α variables). Moreover, then $C|_\rho$ is well-approximated by $\tilde{C}_\rho$ and hence has correlation at most $n^{-\Omega(\gamma)} + n^{-\Omega(\gamma)}$ with parity.

Upper bounding the correlation by 1 for bad restrictions, we see that the overall correlation is at most $n^{-\Omega(\gamma)}$.

We now prove Claim B.3.

Proof of Claim B.3. Let $f_1, \ldots, f_t$ be the LTFs appearing at depth 1 in the circuit. We will divide the analysis based on the fan-ins of the f_is (i.e. the number of variables they depend on).

We denote by β the quantity $\frac{3}{4} + \frac{\varepsilon}{2}$. It can be checked that we have both

$$\beta = \frac{1}{2} + \varepsilon + \frac{\alpha}{2} + \Omega(\gamma) \quad \text{and} \quad 1 - \beta = \frac{\alpha}{2} + \Omega(\gamma). \tag{37}$$

Consider any f_i of fan-in at most n^β. When hit with a random restriction with $*$-probability $n^{-(1-\alpha)}$, we see that the expected number of variables of f_i that survive is at most $n^{\beta-(1-\alpha)} = n^{\alpha-(1-\beta)} = n^{\alpha/2-\Omega(\gamma)}$ by (37) above. By a Chernoff bound, the probability that this number exceeds twice its expectation is exponentially small. Union bounding over all the gates of small fan-in, we see that with probability $1 - \exp(-n^{\Omega(1)})$, all the low fan-in gates depend on at most $2n^{\alpha/2-\Omega(\gamma)}$ many variables after the restriction. We call this high probability event $\mathcal{E}_1$.

Now, we consider the gates of fan-in at least n^β. W.l.o.g., let $f_1, \ldots, f_r$ be these LTFs. Since the total number of wires is at most $n^{1+\varepsilon}$, we have $r \leq n^{1+\varepsilon-\beta} = n^{\frac{1}{2}-\frac{\alpha}{2}-\Omega(\gamma)}$ by (37).

By Theorem 2.11, we know that for any f_i,

$$\mathop{\mathbf{E}}_{\rho}[\mathrm{Var}(f_i|_\rho)] \leq O(\sqrt{p}) = O(\frac{1}{n^{(1-\alpha)/2}}).$$

By linearity of expectation, we have

$$E := \mathop{\mathbf{E}}_{\rho}[\sum_{i=1}^{r} \mathrm{Var}(f_i|_\rho)] \leq O(r \cdot \frac{1}{n^{(1-\alpha)/2}}) = O(n^{(1-\alpha)/2-\Omega(\gamma)} \cdot \frac{1}{n^{(1-\alpha)/2}}) = O(n^{-\Omega(\gamma)}).$$

By Markov's inequality, we see that the probability that $\sum_{i=1}^{r} \mathrm{Var}(f_i|_\rho) > \sqrt{E}$ is at most $\sqrt{E} = n^{-\Omega(\gamma)}$. We let $\mathcal{E}_2$ denote the event that $\sum_{i=1}^{r} \mathrm{Var}(f_i|_\rho) \leq \sqrt{E}$.

Consider the event $\mathcal{E} = \mathcal{E}_1 \wedge \mathcal{E}_2$. A union bound tells us that the probability of $\mathcal{E}$ is at least $1 - n^{-\Omega(\gamma)}$. When this event occurs, we construct the circuit $\tilde{C}_\rho$ from the statement of the claim as follows.

When the event $\mathcal{E}$ occurs, the LTFs of low arity are already $n^{\alpha/2-\Omega(\gamma)}$-juntas, so there is nothing to be done for them.

Now, consider the LTFs of high fan-in, which are $f_1, \ldots, f_r$. For each $f_i|_\rho$ ($i \in [r]$), replace f_i by a bit $b_i \in \{-1, 1\}$ such that $\Pr_x[f_i|_\rho(x) = b_i] \geq \frac{1}{2}$. In the circuit $\tilde{C}_\rho$, these gates thus become constants, which are 0-juntas. The circuit $\tilde{C}_\rho$ now has the required form. We now analyze the error introduced by this operation.

We know that $\Pr_x[f_i|_\rho(x) \neq b_i] \leq 2\mathrm{Var}(f_i|_\rho)$ and thus the overall error introduced is at most $2\sum_{i\in[r]} \mathrm{Var}(f_i|_\rho) \leq O(\sqrt{E}) = n^{-\Omega(\gamma)}$ (since $\mathcal{E}_2$ is assumed to occur). Thus, the circuit $\tilde{C}_\rho$ is an $n^{-\Omega(\gamma)}$-approximation to C. ◀

C Proof of Proposition 4.12

Proof of Proposition 4.12. Let J be the set of variables being set and let $y \in \{-1, 1\}^{|J|}$ denote the random assignment chosen. Let $L_0 = \frac{1}{\varepsilon^2} \cdot 3\log(1/\varepsilon)$. It can be checked that for any $i < L' - L_0$, we have

$$\|w_{>(i+L_0)}\|_2^2 \leq \frac{\varepsilon^2}{9} \cdot \|w_{>i}\|^2 \leq \frac{w_i^2}{9}.$$

Hence, we can choose indices $i_1 = 1, i_2 = 1 + L_0, \cdots, i_{r+2} = 1 + (r+1)L_0 \leq L'$ such that $|w_{i_{j+1}}| \leq \frac{|w_{i_j}|}{3}$ and $\|w_{i_{j+1}}\|_2^2 \leq \frac{\varepsilon^2}{9} \cdot \|w_{i_j}\|_2^2$. Further, we have

$$\sum_{i \notin J} w_i^2 \leq \|w_{>L'}\|^2 \leq \|w_{>i_{r+2}}\|^2 \leq \frac{\varepsilon^2}{9} \cdot \|w_{>i_{r+1}}\|_2^2 \leq \frac{\varepsilon^2}{81} \cdot w_{i_r}^2.$$

We condition on any setting of variables other than $y_{i_1}, \ldots, y_{i_r}$. This means that the constant term of the restricted threshold gate θ' is given by

$$\theta' = \theta'' - \sum_{j\in[r]} w_{i_j} y_{i_j}$$

for some $\theta'' \in \mathbb{R}$. The probability that the threshold gate is not $\frac{1}{\varepsilon}$-imbalanced is at most

$$\begin{aligned}
\Pr_{y_{i_1},\ldots,y_{i_r}}[|\theta'| \leq \frac{1}{\varepsilon^2} \cdot \sqrt{\sum_{i\notin J} w_i^2}] & \\
&\leq \Pr_{y_{i_1},\ldots,y_{i_r}}[|\theta'| \leq \frac{1}{9} \cdot |w_{i_r}|] \\
&= \Pr_{y_{i_1},\ldots,y_{i_r}}[\sum_j w_{i_j} y_{i_j} \in [\theta'' - \frac{1}{9} \cdot |w_{i_r}|, \theta'' + \frac{1}{9} \cdot |w_{i_r}|]]
\end{aligned}$$

Now, as a result of the exponentially decreasing nature of the $|w_{i_j}|$, it follows that for any interval of length at most $|w_{i_r}|/2$, there can be at most one choice of $y_{i_1}, \ldots, y_{i_r}$ such that the $\sum_j w_{i_j} y_{i_j}$ lies in that interval. Thus, we have the given bound. ◀

Strong ETH Breaks With Merlin and Arthur: Short Non-Interactive Proofs of Batch Evaluation

Richard Ryan Williams*

Computer Science Department, Stanford University, Stanford, USA
rrw@cs.stanford.edu

Abstract

We present an efficient proof system for Multipoint Arithmetic Circuit Evaluation: for any arithmetic circuit $C(x_1, \ldots, x_n)$ of size s and degree d over a field $\mathbb{F}$, and any inputs $a_1, \ldots, a_K \in \mathbb{F}^n$,

- the Prover sends the Verifier the values $C(a_1), \ldots, C(a_K) \in \mathbb{F}$ and a proof of $\tilde{O}(K \cdot d)$ length, and
- the Verifier tosses $\text{poly}(\log(dK|\mathbb{F}|/\varepsilon))$ coins and can check the proof in about $\tilde{O}(K \cdot (n+d)+s)$ time, with probability of error less than ε.

For small degree d, this "Merlin-Arthur" proof system (a.k.a. MA-proof system) runs in nearly-linear time, and has many applications. For example, we obtain MA-proof systems that run in c^n time (for various $c < 2$) for the Permanent, #Circuit-SAT for all sublinear-depth circuits, counting Hamiltonian cycles, and infeasibility of 0-1 linear programs. In general, the value of any polynomial in Valiant's class VNP can be certified faster than "exhaustive summation" over all possible assignments. These results strongly refute a Merlin-Arthur Strong ETH and Arthur-Merlin Strong ETH posed by Russell Impagliazzo and others.

We also give a three-round (AMA) proof system for quantified Boolean formulas running in $2^{2n/3+o(n)}$ time, nearly-linear time MA-proof systems for counting orthogonal vectors in a collection and finding Closest Pairs in the Hamming metric, and a MA-proof system running in $n^{k/2+O(1)}$-time for counting k-cliques in graphs.

We point to some potential future directions for refuting the Nondeterministic Strong ETH.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems – Complexity of proof procedures, D.2.4 Software/Program Verification

Keywords and phrases counting complexity, exponential-time hypothesis, interactive proofs, Merlin-Arthur games

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.2

1 Introduction

Suppose you have a circuit of size s that you want to evaluate on k different inputs. In the worst case, you'd expect and need $O(s \cdot k)$ time to do this yourself. What if you asked a powerful computer to evaluate the circuit for you? The computer may be extremely fast relative to you, and send you the k answers almost immediately. But how can you (quickly) check that the computer used *your* circuit, and didn't just make up the answers? Such "delegating/verifiable

* Part of this work was done while visiting the Simons Institute for the Theory of Computing, Berkeley, CA, USA. Supported in part by a David Morgenthaler II Faculty Fellowship, a Sloan Fellowship, and NSF CCF-1212372. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

computation" questions naturally arise in the study of interactive proofs, and have recently seen increased attention in the crypto community (see [24, 21, 18, 6, 20, 48, 34] for a sample of the different models and goals).

For circuits with a certain natural structure[1], we show in this paper how a powerful computer can *very* efficiently prove in one shot (with extremely low probability of error) that its answers are indeed the outputs of your circuit. Omitting low-order terms, the proof is about $\tilde{O}(s+k)$ bits long, and takes about $\tilde{O}(s+k)$ time to verify – roughly proportional to the size of the circuit and the k inputs. The proof system is simple and has no nasty hidden constants, low randomness requirements, and many theoretical applications.

1.1 Our Results

Our evaluation result is best phrased in terms of arithmetic circuits over plus and times gates, evaluated over a finite field. We consider the problem of evaluating such a circuit on many inputs in batch:

▶ **Definition 1.1.** The Multipoint Circuit Evaluation problem: given an arithmetic circuit C on n variables over a finite field $\mathbb{F}$, and a list of inputs $a_1, \ldots, a_K \in \mathbb{F}^n$, output $(C(a_1), \ldots, C(a_K)) \in \mathbb{F}^K$.

An important special case of Multipoint Circuit Evaluation is when the arithmetic circuit is a sum of products of variables (a $\Sigma\Pi$ circuit). This version is called Multivariate Multipoint Evaluation by Kedlaya and Umans [36]; they give the best known algorithms for this case, showing how to solve it in about $(d^n + K)^{1+o(1)}\text{poly}(\log m)$ time over $\mathbb{Z}_m$, where d is the degree of each variable and n is the number of variables. The simplest instance of multipoint evaluation considers circuits that are a sum of products of *one* variable; this case is well-known to have very efficient algorithms (see Section 2). However, for more expressive circuits (such as $\Sigma\Pi\Sigma$, sums of products of sums), no significant improvements over the obvious batch evaluation algorithm have been reported.

Our first result is that multipoint evaluation of *general* arithmetic circuits of low degree can be "delegated" very efficiently, in a publicly verifiable and non-interactive way:

▶ **Theorem 1.2.** *For every finite field $\mathbb{F}$ and $\varepsilon > 0$,* Multipoint Circuit Evaluation *for K points in $\mathbb{F}^n$ on a circuits of n inputs, s gates, and degree d has an probabilistic verifier V where, for every circuit C,*

- *There is a unique proof of $(C(a_1), \ldots, C(a_K))$ that is $\tilde{O}(K \cdot d)$ bits long*[2]*, and*
- *The proof can be verified by V with access to C, $\tilde{O}(1)$ bits of randomness, and $\tilde{O}(K \cdot \max\{d, n\} + s)$ time, such that $(C(a_1), \ldots, C(a_K))$ is output incorrectly with probability at most ε.*

The proof system is fairly simple to motivate. We want the proof to be a succinct representation of the circuit C that is both easy to evaluate on all of the K given inputs, and also easy to verify with randomness. We will set the proof to be a univariate polynomial $Q(x)$ defined over a sufficiently large extension field of $\mathbb{F}$, of degree about $K \cdot d$, that "sketches" the evaluation of the degree-d arithmetic circuit C over all K assignments. The polynomial Q satisfies two conflicting conditions:

[1] In particular, the proof system works for all arithmetic circuits using addition and multiplication over a finite field, where the resulting polynomial has low degree. A surprising number of functions can be efficiently implemented in this way.

[2] The $\tilde{O}$ omits polylog factors in K, $|\mathbb{F}|$, d, s, and $1/\varepsilon$.

1. The verifier can use the sketch Q to efficiently produce the truth table of C. In particular, for some explicitly chosen α_i from the extension of $\mathbb{F}$, $(Q(\alpha_0), Q(\alpha_1), \ldots, Q(\alpha_K)) = (C(a_1), \ldots, C(a_K))$.
2. The verifier can check that Q is a faithful representation of C's behavior on the list of K inputs in about $K + |C|$ time, with randomness.

The construction of Q uses a trick originating from the holographic proofs of Babai *et al.* [9], in which multivariate expressions are efficiently "expressed" as univariate ones. Both of the two items utilize fast algorithms for manipulating univariate polynomials. In the parlance of interactive proofs, Theorem 1.2 gives a *Merlin-Arthur proof system* for batch evaluation (Merlin is the prover, Arthur is the verifier, and Merlin communicates first).

Applications to Some Exponential Time Hypotheses

The results of this paper were originally motivated by attempts to refute *exponential time hypotheses* of increasing strength. The Exponential Time Hypothesis (ETH) [31] is that 3-SAT requires $2^{\varepsilon n}$ time for some $\varepsilon > 0$; ETH has been singularly influential in the area of exact algorithms for NP-hard problems (see [38] for a survey). A more fine-grained version of ETH is the Strong Exponential Time Hypothesis (SETH) [29, 15], which further asserts that k-SAT requires $2^{n-o(n)}$ time for unbounded k. SETH has also been a powerful driver of research in the past several years, especially with its connections to the solvability of basic problems in P (see the recent survey [52]).

Recently, Carmosino *et al.* [16] proposed the Nondeterministic Strong ETH (NSETH): refuting unsatisfiable k-CNFs requires nondeterministic $2^{n-o(n)}$ time for unbounded k. Put another way, NSETH says there are no proof systems that can refute unsatisfiable k-SAT instances significantly more efficiently than enumeration of all variable assignments. The NSETH is quite consistent with known results in proof complexity [42, 11]. Earlier, Carmosino *et al.* (private communication) also proposed a Merlin-Arthur and Arthur-Merlin Strong ETH (MASETH and AMSETH, respectively) which assert that no $O(1)$-round probabilistic proof systems can refute unsatisfiable k-CNFs in $2^{n-\Omega(n)}$ time.

Our first application of Theorem 1.2 is a strong refutation of MASETH and AMSETH:

▶ **Theorem 1.3** (MASETH is False). *There is a probabilistic verifier V where, for every Boolean circuit C on n variables of $o(n)$ depth and bounded fan-in,*

- *There is an $O^\star(2^{n/2})$-bit proof that the number of SAT assignments to C is a claimed value*[3]*, and*
- *The proof can checked by V with access to C, using $O(n)$ bits of randomness and $O^\star(2^{n/2})$ time, with probability of error at most $1/poly(n)$.*

That is, one can refute UNSAT circuits of $2^{o(n)}$ size and $o(n)$ depth significantly faster than brute force enumeration, using a small amount of randomness in verification. Analogues of Theorem 1.3 hold for other #P-complete problems: for instance, the Permanent can be certified in $O^\star(2^{n/2})$ time, and the number of Boolean feasible solutions to a linear program can be certified in $O^\star(2^{3n/4})$. In fact, if we allow the proof to depend on $O(n)$ coins tossed prior to sending the proof, one can also solve *Quantified Boolean Formulas* (QBF) faster:

▶ **Theorem 1.4.** *QBFs with n variables and $m \leq 2^n$ connectives have a three-round $2^{2n/3} \cdot poly(n, m)$ time interactive proof system using $O(n)$ bits of randomness.*

[3] The $O^\star$ notation omits polynomial factors in n.

A seminal result in interactive computation is that PSPACE = IP; that is, polynomial space captures interactive proof systems that use poly(n) time and poly(n) rounds [45]. Theorem 1.4 shows how three rounds of interaction can already significantly reduce the cost of evaluating PSPACE-complete problems. From these results, we see that either $O(n)$ bits of randomness can make a substantial difference in the proof lengths of n-bit propositions, or the Nondeterministic SETH is false. In fact, one can isolate a simple *univariate polynomial identity testing* problem that is solvable in $\tilde{O}(n)$ randomized time and $\tilde{O}(n^2)$ time deterministically, but an $n^{1.999}$-time nondeterministic algorithm would refute NSETH; see Section 3.2.

Applications to Some Polynomial-Time Problems

In Appendix A, we apply Theorem 1.2 to a group of problems at the basis of a recent theory of "hardness within P" [52]. A central problem in this theory is Orthogonal Vectors, which asks if there is an orthogonal pair among n Boolean vectors in d dimensions [54, 43, 57, 13, 3, 4, 10, 2]. The *OV conjecture* is that this problem cannot be solved in $n^{2-\varepsilon} \cdot 2^{o(d)}$, for every $\varepsilon > 0$. It is known that SETH implies the OV conjecture [54, 57]. The OV conjecture can also be refuted in the Merlin-Arthur setting, in the following strong sense:

▶ **Theorem 1.5.** *Let $d \leq n$. There is an MA-proof system such that for every $A \subseteq \{0,1\}^d$ with $|A| = n$, the verifier certifies the number of orthogonal pairs in A, running in $\tilde{O}(n \cdot d)$ time with error probability $1/poly(n)$.*

Because several basic problems in P can be subquadratic-time reduced to Orthogonal Vectors (see the above references and Appendix A), Theorem 1.5 implies subquadratic-time MA-proof systems for these problems as well. To give another example, we also obtain a nearly-linear time proof system for verifying Closest Pairs in the Hamming metric:

▶ **Theorem 1.6.** *Let $d \leq n$. There is an MA-proof system such that for every $A \subseteq \{0,1\}^d$ with $|A| = n$, and every given parameter $k \in \{0, 1, \ldots, d\}$, the verifier certifies for all $v \in A$ the number of points $w \in A$ with Hamming distance at most k from v, running in $\tilde{O}(n \cdot d)$ time with error probability $1/poly(n)$.*

The best known randomized algorithm for computing Hamming closest pairs (as of last year) only runs in $o(n^2)$ time when $d = o(\log^2 n / \log\log n)$ [5]. Finally, we also give an efficient proof system for the k-clique problem:

▶ **Theorem 1.7.** *For every k, there is a MA-proof system such that for every graph G on n nodes, the verifier certifies the number of k-cliques in G using $\tilde{O}(n^{\lfloor k/2 \rfloor + 2})$ time, with error probability $1/poly(n)$.*

2 Preliminaries

For a vector $v \in D^d$ for some domain D, we let $v[i] \in D$ denote the ith component of v. We assume basic familiarity with Computational Complexity, especially the theory of interactive proofs and Merlin-Arthur games as initiated by Goldwasser-Micali-Rackoff [25] and Babai [8] (see Arora and Barak [7], Chapter 8). All of the interactive proofs (also known as "protocols") of this paper will use *public* randomness, visible to the Prover (also known as "Merlin") and the Verifier (also known as "Arthur"). Along the way, we will recall some particulars of known results as needed.

Some Algorithms for Polynomial Computations

We need some classical results in algebraic complexity (see also von zur Gathen and Gerhard [53]). Let $\mathbb{F}$ be an arbitrary field, and let $\text{mult}(n) = O(n \log^2 n)$ be the time needed to multiply two degree-n univariate polynomials.

▶ **Theorem 2.1** (Fast Multipoint Evaluation of Univariate Polynomials [19]). *Given a polynomial $p(x) \in \mathbb{F}[X]$ with $\deg(p) \leq n$, presented as a vector of coefficients $[a_0, \ldots, a_{\deg(p)}]$, and given points $\alpha_1, \ldots, \alpha_n \in \mathbb{F}$, we can output the vector $(p(\alpha_1), \ldots, p(\alpha_n)) \in \mathbb{F}^n$ in $O(\mathit{mult}(n) \cdot \log n)$ additions and multiplications in $\mathbb{F}$.*

▶ **Theorem 2.2** (Fast Univariate Interpolation [28]). *Given a set $\{(\alpha_1, \beta_1), \ldots, (\alpha_n, \beta_n)\} \subset \mathbb{F} \times \mathbb{F}$ with all α_i distinct, we can output the coefficients of $p(x) \in \mathbb{F}[X]$ of degree at most n satisfying $p(\alpha_i) = \beta_i$ for all i, in $O(\mathit{mult}(n) \cdot \log n)$ additions and multiplications in $\mathbb{F}$.*

2.1 More Related Work

Besides what we have already mentioned, there is a vast body of work on non-interactive probabilistic protocols and delegating computation which we are ill-equipped to cover in detail. We confine ourselves to discussing results that seem closest to the present work.[4]

There has been much work on bounding the communication between the prover and verifier. For instance, this is not the first time that Merlin and Arthur have led to an unexpected square-root speedup: Aaronson and Wigderson [1] gave an MA communication protocol for computing the inner product of two n-length vectors which runs in $\tilde{O}(\sqrt{n})$ time. Their protocol uses a nice bivariate encoding of vectors, although it is somewhat different from ours (which is univariate). Gur and Rothblum [27] obtain a similar square-root speedup for checking sums in the "non-interactive property testing" setting. Goldreich and Hastad [22] and Goldreich, Vadhan, and Wigderson [23] studied interactive proofs which seek to minimize the number of bits sent from Merlin to Arthur. The "small bits" case is of course even more restrictive than the "small rounds" case. The latter reference shows that for any language L that has an interactive proof with b bits of communication, there is an $O(1)$-round interactive proof for $\overline{L}$ that uses only $\exp(b)$ communication. The authors also conjectured an "Arthur-Merlin ETH" that #SAT does not have a $2^{o(n)}$-time AM-proof system with $O(1)$ rounds. What we report in this paper is rather far from disproving this "AMETH" conjecture, but it is interesting that some non-trivial progress can be made.

Goldwasser, Kalai, and Rothblum [24] study what they call *delegating computation*, proving (for example) that for all logspace-uniform NC circuits C, one can prove that $C(x) = 1$ on an input x of length n with $\tilde{O}(n)$ verification time, $O(\log n)$ space, and $\text{poly}(\log n)$ communication complexity between the prover and verifier. Despite the amazingly low running time and space usage, the protocols of this work are highly non-interactive: they need $\text{poly}(\log n)$ *rounds* between the prover and verifier as well.

Relating our work to proof complexity, Grochow and Pitassi [26] introduced a new algebraic proof system based on axioms satisfied by any Boolean circuit that solves the polynomial identity testing problem. The proofs in their system can be efficiently verified by running a polynomial identity test, implying they can be viewed as proof of a Merlin-Arthur type. An intriguing property of their proof system is that super-polynomial lower bounds for it would prove lower bounds for the Permanent.

[4] We would be happy to hear of results related to ours that we did not cite.

The area of *verifiable computation* (e.g. [40]) is a new subject in cryptography, and is certainly related to our work. However, in crypto the work appears to be either very specific to particular functions, or it relies on very heavy machinery like probabilistically checkable proofs, or it relies on cryptographic hardness assumptions.

In our setting, we want non-interactive proofs for batch computations that are *shorter* than the computation time, with the typical "perfect completeness" and "low error soundness" conditions preserved, and which work unconditionally.

3 Fast Multipoint Circuit Evaluation (With Merlin and Arthur)

In this section, we give the proof system for multipoint arithmetic circuit evaluation:

▶ **Theorem 3.1.** *For every prime power q and $\varepsilon > 0$,* Multipoint Circuit Evaluation *for K points in $(\mathbb{F}_q)^n$ on an arithmetic circuit C of n inputs, s gates, and degree d has an MA-proof system where:*

- *Merlin sends a proof of $O(K \cdot d \cdot \log(Kqd/\varepsilon))$ bits, and*
- *Arthur tosses at most $\log(Kqd/\varepsilon)$ coins, outputs $(C(a_1), \ldots, C(a_K))$ incorrectly with probability at most ε, and runs in time $(K \cdot \max\{d, n\} + s \cdot \mathit{poly}(\log s)) \cdot \mathit{poly}(\log(Kqd/\varepsilon))$.*

We have stated the theorem at this level of generality because we need good bounds on the parameters to obtain certain consequences. For example, in our proof system for quantified Boolean formulas (Theorem 1.4), the parameters s, K, q, and d are all various exponentials in n.

Because instances of Multipoint Circuit Evaluation have length $O((K \cdot n + s \log s) \cdot \log q)$, the running time of Theorem 3.1 is essentially linear in the input length, up to the factor of d in Merlin's proof (in general, d could be much larger than n). So Theorem 3.1 is extremely powerful for arithmetic circuits of low degree.

Proof. Let q be a prime power and C be an arithmetic circuit over $\mathbb{F}_q$ with degree d, s gates, and n variables. Let $a_1, \ldots, a_K \in \mathbb{F}_q^n$; we want to know $C(a_1), \ldots, C(a_K) \in \mathbb{F}_q$.

Let $\varepsilon > 0$ be arbitrarily small, and let ℓ be the smallest integer such that $q^\ell > (d \cdot K)/\varepsilon$. Let F be the extension field $\mathbb{F}_{q^\ell}$. Note we can construct $\mathbb{F}_{q^\ell}$ rather quickly in the following way: Merlin can send an irreducible polynomial $f(x) \in \mathbb{F}_q[x]$ of degree ℓ, and irreducibility of f can be checked by running Kedlaya-Umans' deterministic irreducibility test in $\ell^{1+o(1)} \log^{2+o(1)} q$ time ([36], Section 8.2).

Since $q^\ell \leq (q \cdot K \cdot d)/\varepsilon$, addition and multiplication in F can be done in $(\log |F|)^{1+o(1)} \leq \log(Kqd/\varepsilon)^{1+o(1)}$ time. Let $S \subseteq F$ be an arbitrary subset of cardinality K. For all $i = 1, \ldots, K$, associate each vector $a_i \in (\mathbb{F}_q)^n$ with a unique element $\alpha_i \in S$, and inversely associate each $\alpha \in S$ with a unique vector $a_\alpha \in (\mathbb{F}_q)^n$. This mapping and its inverse can be easily constructed by listing the first K elements of F under some canonical ordering.

For all $j = 1, \ldots, n$, we define $\Psi_j : F \to F$ as functions satisfying $\Psi_j(\alpha) = a_\alpha[j]$ for every $\alpha \in S$. That is, $\Psi_j(\alpha)$ outputs the jth component of the vector $a_\alpha \in \mathbb{F}_q^n$ associated with $\alpha \in S$. Since each Ψ_j is defined by K input/output pairs, the Ψ_j can be instantiated as polynomials of degree at most K. By efficient polynomial interpolation (Theorem 2.2), the degree-K polynomials $\Psi_j(x) \in F[x]$ for all $j = 1, \ldots, n$ can be constructed in $n \cdot K \cdot \mathrm{poly}(\log K)$ additions and multiplications.

Define the univariate polynomial $R(x) := C(\Psi_1(x), \ldots, \Psi_n(x))$ over F. By the construction of Ψ_j, we see that for all $i = 1, \ldots, K$, $R(\alpha_i) = C(\Psi_1(\alpha_i), \ldots, \Psi_n(\alpha_i)) = C(a_i[1], \ldots, a_i[n]) = C(a_i)$. Furthermore, $\deg(R) \leq \deg(C) \cdot (\max_j \Psi_j) \leq d \cdot K$.

Now we describe the protocol.

1. Merlin sends the coefficients of a polynomial $Q(x)$ over F of degree at most $d \cdot K$, encoded in $d \cdot K \cdot \log(|F|)$ bits. Merlin claims that $Q(x) = R(x)$, as defined above.
2. Arthur picks a uniform random $r \in F$ (taking at most $\log(Kqd/\varepsilon)$ bits to describe), and wishes to check that

   $$Q(r) = R(r) := C(\Psi_1(r), \ldots, \Psi_n(r)),$$

   over F. Evaluating $Q(r)$ takes $d \cdot K \cdot (\log(Kqd/\varepsilon))^{1+o(1)}$ time, by Horner's method. We claim that $R(r)$ can be computed in $(K \cdot n + s) \cdot (\log |F|)^{1+o(1)}$ time. First, the n polynomials Ψ_j of degree K can be constructed in $n \cdot K \cdot \text{poly}(\log K)$ additions and multiplications (as described above). Given the coefficients of the Ψ_j polynomials, computing all values $v_j := \Psi_j(r)$ can be done straightforwardly in $O(K \cdot n)$ additions and multiplications, by producing the powers $r^0, r^1, \ldots, r^K$ and then computing n linear combinations of these powers. (Note that each resulting value v_j takes $O(\log |F|) \leq \text{poly}(Kqd/\varepsilon)$ bits to represent.) Then Arthur computes $C(v_1, \ldots, v_n)$ in $s \cdot \text{poly}(\log s)$ additions and multiplications, by simple circuit evaluation over F. The total running time is $(K \cdot n + s \cdot \text{poly}(\log s)) \cdot (\log |F|)^{1+o(1)}$.
3. Arthur *rejects the proof* if $Q(r) \neq C(v_1, \ldots, v_n)$; otherwise, he uses univariate multipoint evaluation (Theorem 2.1) to compute $(Q(\alpha_1), \ldots, Q(\alpha_K))$, in $K \cdot d \cdot \text{poly}(\log(Kd)) \cdot (\log |F|)^{1+o(1)}$ time.

   On the one hand, if Merlin sends $Q(x) := R(x)$, then Arthur always outputs the tuple

   $$(R(\alpha_1), \ldots, R(\alpha_K)) = (C(a_1), \ldots, C(a_n)),$$

regardless of the $r \in F$ chosen. On the other, if Merlin sends a "bad" polynomial $Q(x) \neq R(x)$ and Arthur fails to pick an $r \in F$ such that $Q(r) \neq R(r)$, then Merlin may convince Arthur of an incorrect K-tuple $(Q(\alpha_1), \ldots, Q(\alpha_K))$. However, since the degrees of Q and R are both at most $d \cdot K$, this failure of Arthur occurs with probability at most $(d \cdot K)/q^\ell < \varepsilon$. ◀

3.1 Evaluating Sums Over Polynomials

The multipoint evaluation protocol of Theorem 3.1 can be applied to perform a one-round "sum-check" faster than the obvious algorithm:

▶ **Theorem 3.2.** *Given a prime p, an $\varepsilon > 0$, and an arithmetic circuit C with degree d, $s \geq n$ gates, and n variables, the sum*

$$\sum_{(b_1,\ldots,b_n) \in \{0,1\}^n} C(b_1, \ldots, b_n) \bmod p$$

can be computed by a Merlin-Arthur protocol running in $2^{n/2} \cdot \text{poly}(n, s, d, \log(p/\varepsilon))$ time tossing only $n/2 + O(\log(pd/\varepsilon))$ coins, with probability of error ε.

As it applies Theorem 3.1 in a straightforward way, the proof of Theorem 3.2 in fact works for any finite field. Therefore, every polynomial over a finite field in the class VNP ([49, 51]) has a MA-proof system that beats exhaustive search in a strong sense.

Proof. (of Theorem 3.2) For simplicity, assume n is even. Given an arithmetic circuit C for which we wish to evaluate its sum over all Boolean inputs, define the $n/2$-variable circuit

$$C'(x_1, \ldots, x_{n/2}) := \sum_{(b_1,\ldots,b_{n/2}) \in \{0,1\}^{n/2}} C(x_1, \ldots, x_{n/2}, b_1, \ldots, b_{n/2}).$$

Note that $\deg(C') = d$ and $\text{size}(C') \leq 2^{n/2} \cdot s$. In order to compute the full sum of $C(b_1, \ldots, b_n)$ over all 2^n Boolean points, it suffices to evaluate C' on all of its $K := 2^{n/2}$ Boolean points $a_1, \ldots, a_K \in \{0,1\}^n$.

Applying the batch evaluation protocol of Theorem 3.1, there is an MA-proof system where Merlin sends a proof of $2^{n/2} \cdot d \cdot \text{poly}(n, \log(pd/\varepsilon))$ bits, then Arthur tosses $n/2 + \log(pd/\varepsilon)$ coins, runs in $(2^{n/2} \cdot \max\{n, d\} + 2^{n/2} \cdot s \cdot \text{poly}(\log s)) \cdot \text{poly}(n, \log(pd/\varepsilon))$ time, and outputs $(C'(a_1), \ldots, C(a_{2^{n/2}}))$ incorrectly with probability at most ε. The result follows. ◀

Two important corollaries of Theorem 3.2 are $O^\star(2^{n/2})$-time proof systems for the Permanent and #SAT problems. The result for Permanent follows immediately from Ryser's formula [44], which shows that the permanent of any $n \times n$ matrix M can be written in the form

$$\sum_{(a_1,\ldots,a_n)\in\{0,1\}^n} C_M(a_1, \ldots, a_n),$$

where C_M is a $\text{poly}(n)$-size arithmetic circuit of degree $O(n)$ that can be determined from M in $\text{poly}(n)$ time. We describe the *#SAT* protocol in detail:

▶ **Theorem 3.3.** *For any $k > 0$, #SAT for Boolean formulas with n variables and m connectives has an MA-proof system using $2^{n/2} \cdot \text{poly}(n, m)$ time with randomness $O(n)$ and error probability $1/\exp(n)$.*

Proof. Let F be a Boolean formula over AND, OR, and NOT with n variables and m connectives. First, any Boolean formula F can be "re-balanced" as in the classical results of Brent [12] and Spira [47], obtaining in $\text{poly}(m)$ time a formula F' equivalent to F, where F' has depth at most $c \log m$ and at most m^c connectives for some constant $c > 0$.

Next, we replace each AND, OR, and NOT gate of F' with an equivalent polynomial of degree 2, by the usual "arithmetization." More precisely, each $OR(x, y)$ is replaced with $x+y-x\cdot y$, each $AND(x, y)$ is replaced with $x \cdot y$, and each $NOT(1-x)$ is replaced with $1-x$. The resulting arithmetic formula $P(x_1, \ldots, x_n)$ computes $F'(b_1, \ldots, b_n) = P(b_1, \ldots, b_n)$ for every $(b_1, \ldots, b_n) \in \{0,1\}^n$. Furthermore, due to the re-balancing step and the fact that every gate has outdegree 1, we have $\deg(P) \leq 2^{c \log m} \leq m^{O(1)}$ (note the worst case is when every gate is an AND).

Set $p > 2^n$ to be prime; note by Bertrand's postulate we may assume $p < 2^{n+1}$. We can always find such a prime deterministically in $2^{n/2+o(n)}$ time by an algorithm of Lagarias and Odlyzko [37]. (Alternatively, the prover could send p to the verifier, along with a deterministically verifiable $\text{poly}(n)$-length proof of primality [41].) Then F' has exactly r satisfying assignments if and only if

$$\sum_{(b_1,\ldots,b_n)\in\{0,1\}^n} P(b_1, \ldots, b_n) = r \bmod p.$$

Since $\deg(P) \leq m^{O(1)}$, we can apply Theorem 3.2 directly and obtain the result. ◀

Another corollary of Theorem 3.2 is that Merlin and Arthur can also count Hamiltonian cycles in n-node graphs in $O^\star(2^{n/2})$ time, by construing the inclusion-exclusion method of Karp [35] running in $O^\star(2^n)$ time as a sum over 2^n Boolean values on an arithmetic circuit of $\text{poly}(n)$ size. In particular, Karp's algorithm works by counting the n-step walks in a graph, then subtracting the count of n-step walks that miss at least one node, adding back the count of n-step walks that miss at least two nodes, etc. Each of these counts is computable by a single arithmetic circuit $C(y_1, \ldots, y_n)$ of $O(n^4)$ size which, on the input $y \in \{0,1\}^n$,

counts the n-step walks over the subgraph of G defined by the vector y (negating the count if y has an odd number of zeroes).

Theorem 3.3 shows that Merlin and Arthur can count the number of satisfying assignments to Boolean formulas of $2^{\delta n}$ size in $2^{n(1/2+O(\delta))}$ time. It also immediately follows from Theorem 3.3 that we can solve #SAT on bounded fan-in circuits of depth $o(n)$ in $2^{n/2+o(n)}$ time, as such circuits can always be expressed as formulas of $\exp(o(n))$ size. It is also clear from the proof that we can trade off proof length and verification time: if we restrict the proofs to have length $2^\ell \leq 2^{n/2}$ (so that Merlin sends a polynomial of degree roughly 2^ℓ), then verifying the remaining sum over $n-\ell$ variables takes $O^\star(2^{n-\ell})$ time.

We also observe that with more rounds of interaction, Merlin and Arthur can use shorter proofs. This is somewhat expected, because it is well-known that in $O(n)$ rounds, we can compute #SAT with $\mathrm{poly}(n)$ communication and $\mathrm{poly}(n)$ verification time [39].

▶ **Theorem 3.4.** *For any $k > 0$, and $c > 2$, #SAT for Boolean formulas with n variables and m connectives has an interactive proof system with c rounds of interaction, using $2^{n/(c+1)} \cdot \mathrm{poly}(n, m)$ time with randomness $O(n)$ and error probability $1/\exp(n)$.*

Proof. (Sketch) We essentially interpolate between our protocol and the LFKN protocol ([39]) for $\#SAT$. Let F be a Boolean formula over AND, OR, and NOT with n variables and m connectives, and let P be its arithmetization as in Theorem 3.3. We will work modulo a prime $p > 2^n$, as before. For simplicity let us assume n is divisible by $c+1$, and that $m \leq 2^{o(n)}$. Partition the set of variables into subsets $S_1, \ldots, S_{c+1}$ of $n/(c+1)$ variables each. Via interpolation, define the polynomials $\Psi_1, \ldots, \Psi_{\frac{n}{c+1}}$ analogously to Theorem 3.1, where for all $j \in \{0, 1, \ldots, 2^{n/(c+1)} - 1\}$, $\Psi_i(j)$ outputs the ith bit of the j in $n/(c+1)$-bit binary representation. Now consider the polynomial in $c+1$ variables:

$$Q_1(y) := \sum_{\substack{j_2,\ldots,j_{c+1} \\ \in\{0,1,\ldots,2^{n/(c+1)}-1\}}} P(\Psi_1(y), \ldots, \Psi_{\frac{n}{c+1}}(y), \Psi_1(j_2), \ldots, \Psi_{\frac{n}{c+1}}(j_2), \ldots\ldots, \Psi_{\frac{n}{c+1}}(j_{c+1})).$$

In the first round of interaction, an honest prover sends $Q_1(y)$, which has degree $2^{n/(c+1)+o(n)}$. The verifier then chooses a random $r_1 \in \mathbb{F}_p$, and sums $Q_1(y)$ over all points $\{0, 1, \ldots, 2^{n/(c+1)} - 1\}$.

In the kth round of interaction for $k = 2, \ldots, c$, an honest prover sends the $2^{n/(c+1)+o(n)}$-degree polynomial $Q_k(y)$, which is

$$\sum_{\substack{j_{k+1},\ldots,j_{c+1} \\ \in\{0,\ldots,2^{n/(c+1)}-1\}}} P(\Psi_1(r_1), \ldots, \Psi_{\frac{n}{c+1}}(r_{k-1}), \Psi_1(y), \ldots, \Psi_{\frac{n}{c+1}}(y), \Psi_1(j_{c+1}) \ldots, \Psi_{\frac{n}{c+1}}(j_{c+1})).$$

The verifier again chooses a random $r_k \in \mathbb{F}_p$.

Finally in the cth round, after the prover has sendt $Q_c(y)$ and the verifier has chosen $r_c \in \mathbb{F}_p$ at random, the remaining computation is to compute the sum $\sum_{i=0}^{2^{n/(c+1)}-1} Q_c(j_i)$, and to verify that $Q_c(r_c)$ equals

$$\sum_{\substack{j_{c+1} \\ \in\{0,\ldots,2^{n/(c+1)}-1\}}} P(\Psi_1(r_1), \ldots, \Psi_{\frac{n}{c+1}}(r_1), \ldots, \Psi_1(r_c), \ldots, \Psi_{\frac{n}{c+1}}(r_c), \Psi_1(j_{c+1}), \ldots, \Psi_{\frac{n}{c+1}}(j_{c+1})).$$

In each of the c rounds, the probability of picking a "bad" r_i is at most $2^{\frac{n}{c+1}+o(n)}/p \leq \exp(-\Omega(n))$. ◀

Thus, with $\omega(1)$ rounds of interaction, Arthur and Merlin can compute $\#SAT$ in $2^{o(n)}$ verification time and communication.

3.2 Univariate Polynomial Identity Testing and the Nondeterministic SETH

A nice aspect of Theorem 3.2 and its corollaries is that the randomness is low: for example, the obvious derandomization strategy of simulating all $O^\star(2^{n/2})$ coin tosses recovers a nondeterministic $O^\star(2^n)$ time algorithm for counting SAT assignments modulo 2.

The proof system itself motivates the following problem. Let *univariate polynomial identity testing* (UPIT) be the problem of testing identity for two arithmetic circuits with *one* variable, degree n, and $O(n)$ wires, over a field of order $\text{poly}(n)$. The following corollary is immediate from the proofs of Theorems 1.2, 3.3, and the above observations:

▶ **Corollary 3.5.** *If UPIT* $\in \mathsf{NTIME}[n^{2-\varepsilon}]$ *for some* $\varepsilon > 0$*, then #Circuit-SAT for* $o(n)$*-depth circuits is computable in nondeterministic* $2^{n(1-\varepsilon/2)+o(n)}$ *time. By [56, 32, 16], this further implies that* E^{NP} *does not have* $2^{o(n)}$*-size sublinear-depth circuits.*

In particular, the randomized verification task of Arthur in the protocol of Theorem 3.3 directly reduces to solving UPIT on two univariate circuits of degree $2^{n/2+o(n)}$ and size $2^{n/2+o(n)}$. Hence, assuming the hypothesis of Corollary 3.5, Arthur's verification can be performed deterministically in $2^{n(1-\varepsilon/2)+o(n)}$ time.

This is an intriguing example of how derandomization *within polynomial time* can imply strong circuit lower bounds: it is easy to see that UPIT is solvable in $\tilde{O}(n)$ time with randomness, and in $\tilde{O}(n^2)$ time deterministically, by efficient interpolation on $n+1$ distinct points (Theorem 2.2). In all other cases we are aware of (such as [33, 55]), the necessary derandomization problem is only known to be solvable in deterministic *exponential* time. Thus, the Nondeterministic SETH predicts that the exponent of the simple $\tilde{O}(n^2)$ algorithm for UPIT cannot be improved, even with nondeterminism.

4 Quantified Boolean Formulas

In the previous section, we saw how generic #P counting problems can be certified faster than exhaustive search. We can also give less-than-2^n time three-round proof systems for certifying quantified Boolean formulas, a PSPACE-complete problem. Our quantified Boolean formulas have the form

$$(Q_1 x_1) \cdots (Q_n x_n) F(x_1, \ldots, x_n),$$

where F is an arbitrary propositional formula on m connectives, and each $Q_i \in \{\exists, \forall\}$.

Reminder of Theorem 1.4 Quantified Boolean Formulas with n variables and $m \leq 2^{o(n)}$ connectives have a three-round interactive proof system running in $2^{2n/3} \cdot \text{poly}(n, m)$ time with $O(n)$ bits of randomness.

Proof. Let $\phi = (Q_1 x_1) \cdots (Q_n x_n) F(x_1, \ldots, x_n)$ be a quantified Boolean formula to certify. Let $\delta > 0$ be a parameter to set later. First, convert the propositional formula F' to an equivalent arithmetic circuit P of $\text{poly}(m)$ degree and size, as in Theorem 3.3. Note that P outputs 0 or 1 on every Boolean input to its variables. Next, determine whether the quantifier suffix $(Q_{n-\delta n+1} x_{n-\delta n+1}) \cdots (Q_n x_n)$ contains at least as many existential quantifiers as universal quantifiers.

Case 1. If there are more existentially quantified variables, convert the subformula

$$\phi'(x_1, \ldots, x_{n-\delta n}) = (Q_{n-\delta n+1} x_{n-\delta n+1}) \cdots (Q_n x_n) P(x_1, \ldots, x_n)$$

into an arithmetic formula P' in a standard way, where each $(\exists x_i)$ is replaced by a sum over $x_i \in \{0,1\}$, and each $(\forall x_i)$ is replaced by a product over $x_i \in \{0,1\}$. The formula P' has size $2^{\delta n} \cdot \text{poly}(m)$, for the tree of possible assignments to the last $2^{\delta n}$ variables times the size of the polynomial P.

It is easy to see that $P'(a_1, \ldots, a_{n-\delta n})$ is nonzero (over $\mathbb{Z}$) on a Boolean assignment $(a_1, \ldots, a_{n-\delta n})$ if and only if $\phi'(a_1, \ldots, a_{n-\delta n})$ is true. Moreover, P' has degree at most $\text{poly}(m) \cdot 2^{\delta n/2}$, since there are most $\delta n/2$ universal quantifiers among the last δn variables (so the $2^{\delta n}$ tree contains at most $\delta n/2$ layers of multiplication gates). Note the value $V_{a_1,\ldots,a_{n-\delta n}} = P'(a_1, \ldots, a_{n-\delta n})$ is always at most $(2^n \cdot m)^{O(2^{\delta n/2})}$.

Our protocol begins by having Arthur send a random prime p from the interval $[2, 2^{2n^2} \cdot m]$ to Merlin, to help reduce the size of the values $V_{a_1,\ldots,a_{n-\delta n}}$. (A similar step also occurs in the proof that IP = PSPACE [45, 46].) Since a nonzero $V_{a_1,\ldots,a_{n-\delta n}}$ has at most $O(2^{\delta n/2} nm)$ prime factors, the probability that a random $p \in [2, 2^{2n^2} \cdot m]$ divides a fixed $V_{a_1,\ldots,a_{n-\delta n}}$ is at most

$$\frac{O(2^{\delta n/2} n(n + \log m))}{2^{2n^2}},$$

by the Prime Number Theorem. By the union bound, p divides $V_{a_1,\ldots,a_{n-\delta n}}$ for some $a_1, \ldots, a_{n-\delta n} \in \{0,1\}$ with probability at most $(\log m)/2^{\Omega(n^2)}$.

Therefore for all $a_1, \ldots, a_{n-\delta n} \in \{0,1\}$, the "non-zeroness" of $P'(a_1, \ldots, a_{n-\delta n})$ over $\mathbb{Z}$ is preserved over the field $\mathbb{F}_p$, with high probability. Merlin and Arthur will work over $\mathbb{F}_p$ in the following.

Applying Theorem 3.1 to P' with $d := \text{poly}(m) \cdot 2^{\delta n/2}$, $p := 2^{2n} \cdot m$, $K := 2^{n-\delta n}$, and $s := \text{poly}(m) \cdot 2^{\delta n}$, there is an MA-proof system where Merlin sends a proof of length at most $2^{n-\delta n/2} \cdot \text{poly}(n)$ bits, while Arthur uses at most $\text{poly}(n)$ coins and $(2^{n-\delta n/2} + 2^{\delta n}) \cdot \text{poly}(n, m)$ time, outputting the value of P' on all $2^{n-\delta n}$ Boolean inputs with high probability. It is easy to determine the truth value of the original QBF ϕ from the $2^{n-\delta n}$-length truth table of P'; this is simply a formula evaluation on an $O(2^{n-\delta n})$-size formula defined by the quantifier prefix $(Q_1 x_1) \cdots (Q_{n-\delta n} x_{n-\delta n})$.

Setting $\delta = 2/3$ yields a $2^{2n/3} \cdot \text{poly}(n, m)$-length proof and an analogous running time bound.

Case 2. If there are at least as many universal variables as existential ones, then Merlin and Arthur decide to prove that $\neg\phi$ is false, by flipping the type of every quantifier (from existential to universal, and vice-versa) and replacing P with an arithmetic circuit for $\neg F$. Now the quantifier suffix $(Q'_{n-\delta n+1} x_{n-\delta n+1}) \cdots (Q'_n x_n)$ of the new QBF contains more existential quantifiers than universal ones, and we proceed as in the first case, evaluating an $(n - \delta n)$-variable formula of $2^{\delta n}$ size (and at most $\delta n/2$ universally quantified variables) on all of its possible assignments, and inferring the truth or falsity of the QBF from that evaluation. ◀

5 Conclusion

By a simple but powerful protocol for batch multipoint evaluation, we have seen how non-interactive proof systems can be exponentially more powerful than randomized or

nondeterministic algorithms, assuming some exponential-time hypotheses. There are many questions left to pursue, for instance:

- Are there more efficient proof systems if we just want to prove that a formula is UNSAT? Perhaps UNSAT has an MA-proof system of $O^\star(2^{n/3})$ time. Perhaps Parity-SAT could be certified more efficiently, exploiting the nice properties of characteristic-two fields? By the Valiant-Vazirani lemma [50], this would imply a three-round interactive proof system for UNSAT that is also more efficient. Our MA-proof systems all have extremely low randomness requirements of Arthur. If we allowed $2^{\delta n}$ bits of randomness for some $\delta > 0$, perhaps they can be improved further.
- Faster nondeterministic UNSAT algorithms are by now well-known to imply circuit lower bounds for problems in nondeterministic exponential time [55, 32]. Can the proof systems of this paper be applied to conclude new lower bounds? One difficulty is that we already know $\mathsf{MAEXP} \not\subset \mathsf{P/poly}$ [14]. More seriously, it seems possible that one could apply our protocol for #SAT on circuits of $o(n)$ depth to show that (for instance) $\mathsf{E}^{\mathsf{promiseMA}}$ does not have $2^{o(n)}$ size formulas; this would be a major advance in our understanding of exponential-size circuits.
- Can $O^\star(2^{n/2})$-time Merlin-Arthur proof system for $\#SAT$ be converted into a construction of nondeterministic circuits of $(2-\varepsilon)^n$ size for UNSAT? To do this, we would want to have a small collection of coin tosses that suffices for verification. If we convert the proof system into an Arthur-Merlin game in the standard way, the protocol has the following structure: for a proof-length parameter ℓ, we can toss $O(\ell \cdot n)$ random coins are tossed prior to the proof, then Merlin can give a single $\tilde{O}(\ell)$-bit proof of the protocol that needs to be simulated on $O(\ell)$ different coin tosses of $n/2 + \tilde{O}(1)$ bits each. The difficulty is that each of these $O(\ell)$ coin tosses takes $\Omega(2^n/\ell)$ time for Arthur to verify on his own, as far as we can tell. So even though the probability of error here could be extremely small (less than $1/2^{\Omega(\ell)}$) we do not know how to get a $(2-\varepsilon)^n$ time algorithm for verification.
- Does QBF on n variables and $\mathrm{poly}(n)$ connectives have an MA-proof system using $(2-\varepsilon)^n$ time, for some $\varepsilon > 0$?

Acknowledgements. I thank Russell Impagliazzo for sending a draft of his paper (with coauthors) on NSETH, MASETH, and AMSETH, and for discussions on the #SAT protocol. I also thank Petteri Kaski for suggesting that I add a protocol for Closest Pair and Hamiltonian Cycle, Shafi Goldwasser for references, and the anonymous reviewers for their comments.

References

1 Scott Aaronson and Avi Wigderson. Algebrization: A new barrier in complexity theory. *ACM TOCT*, 1, 2009.

2 Amir Abboud, Arturs Backurs, and Virginia V. Williams. Quadratic-time hardness of LCS and other sequence similarity measures. In *FOCS*, pages 59–78, 2015.

3 Amir Abboud and Virginia Vassilevska Williams. Popular conjectures imply strong lower bounds for dynamic problems. In *FOCS*, pages 434–443, 2014.

4 Amir Abboud, Richard Ryan Williams, and Huacheng Yu. More applications of the polynomial method to algorithm design. In *SODA*, pages 218–230, 2015.

5 Josh Alman and Ryan Williams. Probabilistic polynomials and hamming nearest neighbors. In *FOCS*, pages 136–150, 2015.

6 Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. From secrecy to soundness: Efficient verification via secure computation. In *Proc. ICALP, Part I*, pages 152–163, 2010.

7 Sanjeev Arora and Boaz Barak. *Computational Complexity – A Modern Approach*. Cambridge University Press, 2009.

8 László Babai. Trading group theory for randomness. In *STOC*, pages 421–429, 1985.

9 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*, pages 21–32, 1991.

10 Arturs Backurs and Piotr Indyk. Edit distance cannot be computed in strongly subquadratic time (unless SETH is false). In *STOC*, pages 51–58, 2015.

11 Christopher Beck and Russell Impagliazzo. Strong ETH holds for regular resolution. In *STOC*, pages 487–494, 2013.

12 Richard P. Brent. The parallel evaluation of general arithmetic expressions. *J. ACM*, 21(2):201–206, 1974.

13 Karl Bringmann. Why walking the dog takes time: Frechet distance has no strongly subquadratic algorithms unless SETH fails. In *FOCS*, pages 661–670, 2014.

14 Harry Buhrman, Lance Fortnow, and Thomas Thierauf. Nonrelativizing separations. In *CCC*, pages 8–12, 1998.

15 Chris Calabro, Russell Impagliazzo, and Ramamohan Paturi. The complexity of satisfiability of small depth circuits. In *Parameterized and Exact Complexity (IWPEC)*, pages 75–85, 2009.

16 Marco Carmosino, Jiawei Gao, Russell Impagliazzo, Ivan Mikhailin, Ramamohan Paturi, and Stefan Schneider. Nondeterministic extensions of the Strong Exponential Time Hypothesis and consequences for non-reducibility. In *Proceedings of the ACM Conference on Innovations in Theoretical Computer Science, (ITCS)*, pages 261–270, 2016.

17 Timothy M. Chan and Ryan Williams. Deterministic APSP, Orthogonal Vectors, and more: Quickly derandomizing Razborov-Smolensky. In *SODA*, pages 1246–1255, 2016.

18 Kai-Min Chung, Yael Tauman Kalai, and Salil P. Vadhan. Improved delegation of computation using fully homomorphic encryption. In *CRYPTO*, pages 483–501, 2010.

19 Charles M. Fiduccia. Polynomial evaluation via the division algorithm: The fast fourier transform revisited. In *STOC*, pages 88–93, 1972.

20 Dario Fiore and Rosario Gennaro. Publicly verifiable delegation of large polynomials and matrix computations, with applications. In *ACM CCS*, pages 501–512, 2012.

21 Rosario Gennaro, Craig Gentry, and Bryan Parno. Non-interactive verifiable computing: Outsourcing computation to untrusted workers. In *CRYPTO*, pages 465–482, 2010.

22 Oded Goldreich and Johan Håstad. On the complexity of interactive proofs with bounded communication. *Inf. Process. Lett.*, 67(4):205–214, 1998.

23 Oded Goldreich, Salil P. Vadhan, and Avi Wigderson. On interactive proofs with a laconic prover. *Computational Complexity*, 11(1-2):1–53, 2002.

24 Shafi Goldwasser, Yael Tauman Kalai, and Guy N. Rothblum. Delegating computation: Interactive proofs for muggles. *J. ACM*, 62(4):27, 2015. Original in STOC'08.

25 Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof-systems (extended abstract). In *STOC*, pages 291–304, 1985.

26 Joshua A. Grochow and Toniann Pitassi. Circuit complexity, proof complexity, and polynomial identity testing. In *FOCS*, pages 110–119, 2014.

27 Tom Gur and Ron D. Rothblum. Non-interactive proofs of proximity. In *Proceedings of the ACM Conference on Innovations in Theoretical Computer Science, (ITCS)*, pages 133–142, 2015.

28 Ellis Horowitz. A fast method for interpolation using preconditioning. *Inf. Process. Lett.*, 1(4):157–163, 1972.

29 Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-SAT. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001.

30 Russell Impagliazzo, Ramamohan Paturi, and Stefan Schneider. A satisfiability algorithm for sparse depth two threshold circuits. In *FOCS*, pages 479–488, 2013.

31 Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001.

32 Hamid Jahanjou, Eric Miles, and Emanuele Viola. Local reductions. In *ICALP*, pages 749–760, 2015.

33 Valentine Kabanets and Russell Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Computational Complexity*, 13(1-2):1–46, 2004.

34 Yael Tauman Kalai, Ran Raz, and Ron D. Rothblum. How to delegate computations: the power of no-signaling proofs. In *STOC*, pages 485–494, 2014.

35 Richard M Karp. Dynamic programming meets the principle of inclusion and exclusion. *Operations Research Letters*, 1(2):49–51, 1982.

36 Kiran S. Kedlaya and Christopher Umans. Fast polynomial factorization and modular composition. *SIAM J. Comput.*, 40(6):1767–1802, 2011.

37 J. C. Lagarias and Andrew M. Odlyzko. Computing pi(x): An analytic method. *J. Algorithms*, 8(2):173–191, 1987.

38 Daniel Lokshtanov, Dániel Marx, and Saket Saurabh. Lower bounds based on the exponential time hypothesis. *Bulletin of the EATCS*, 105:41–72, 2011.

39 Carsten Lund, Lance Fortnow, Howard Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. *JACM*, 39(4):859–868, 1992.

40 Bryan Parno, Jon Howell, Craig Gentry, and Mariana Raykova. Pinocchio: Nearly practical verifiable computation. In *IEEE Symposium on Security and Privacy*, pages 238–252, 2013.

41 Vaughan R. Pratt. Every prime has a succinct certificate. *SIAM J. Comput.*, 4(3):214–220, 1975.

42 Pavel Pudlák and Russell Impagliazzo. A lower bound for DLL algorithms for k-sat (preliminary version). In *SODA*, pages 128–136, 2000.

43 Liam Roditty and Virginia Vassilevska Williams. Fast approximation algorithms for the diameter and radius of sparse graphs. In *STOC*, pages 515–524, 2013.

44 Herbert John Ryser. *Combinatorial mathematics*. Mathematical Association of America, 1963. The Carus mathematical monographs.

45 Adi Shamir. IP = PSPACE. *Journal of the Association for Computing Machinery*, 39(4):869–877, 1992.

46 A. Shen. IP = PSPACE: simplified proof. *J. ACM*, 39(4):878–880, 1992.

47 P. M. Spira. On time hardware complexity tradeoffs for boolean functions. In *Proceedings of the Fourth Hawaii International Symposium on System Sciences*, pages 525–527, 1971.

48 Justin Thaler. Time-optimal interactive proofs for circuit evaluation. In *CRYPTO*, pages 71–89, 2013.

49 Leslie Valiant. Completeness classes in algebra. In *STOC*, pages 249–261, 1979.

50 Leslie Valiant and Vijay Vazirani. NP is as easy as detecting unique solutions. *Theor. Comp. Sci.*, 47(3):85–93, 1986.

51 Leslie G. Valiant. Reducibility by algebraic projections. In *Logic and Algorithmic: an International Symposium held in honor of Ernst Specker*, volume 30, pages 365–380, 1982. Monogr. No. 30 de l'Enseign. Math.

52 Virginia Vassilevska Williams. Hardness of easy problems: Basing hardness on popular conjectures such as the strong exponential time hypothesis. In *Proc. International Symposium on Parameterized and Exact Computation*, pages 16–28, 2015.

53 Joachim von zur Gathen and Jürgen Gerhard. *Modern Computer Algebra*. Cambridge University Press, 2013. 3rd edition.

54 Ryan Williams. A new algorithm for optimal 2-constraint satisfaction and its implications. *Theor. Comput. Sci.*, 348(2-3):357–365, 2005.

55 Ryan Williams. Improving exhaustive search implies superpolynomial lower bounds. *SIAM Journal on Computing*, 42(3):1218–1244, 2013. See also STOC'10.

56 Ryan Williams. Non-uniform ACC circuit lower bounds. *J. ACM*, 61(1):2:1–2:32, 2014. Preliminary version in CCC'11.

57 Ryan Williams and Huacheng Yu. Finding orthogonal vectors in discrete structures. In *SODA*, pages 1867–1877, 2014.

A Quick Proof Systems For Some Poly-Time Problems

We can also obtain nearly-linear time MA-proof systems for quite a few problems which have been conjectured to be hard to solve faster than quadratic time. Perhaps the most illustrative example is a proof system for computing orthogonal pairs of vectors. Via reductions, this result implies analogous proof systems for several other quadratic-time solvable problems (see [4]); we omit the details here.

▶ **Theorem A.1.** *Let $d \leq n$. For every $A \subseteq \{0,1\}^d$ such that $|A| = n$, there is a MA-proof system certifying for every $v \in A$ if there is a $u \in A$ such that $\langle v, u\rangle = 0$, with $\tilde{O}(n \cdot d)$ time and error probability $1/poly(n)$.*

Proof. Let p be a prime greater than $n^2 \cdot d$. Define the $2d$-variable polynomial

$$P(x_1, \ldots, x_d, y_1, \ldots, y_d) := \prod_{i=1}^{d} (1 - x_i \cdot y_i)\,.$$

Observe $\deg(P) \leq 2d$, and for a pair of Boolean vectors $u, v \in \{0,1\}^d$, $P(u,v) = 1$ if $\langle u, v\rangle = 0$, otherwise $P(u,v) = 0$. Then, the polynomial

$$P'(u[1], \ldots, u[d]) := \sum_{j=1,\ldots,n} P(u[1], \ldots, u[d], v_j[1], \ldots, v_j[d])$$

counts the number of vectors in A that are orthogonal to the input vector $u \in \{0,1\}^d$. Note the size of P' as an arithmetic circuit is $O(n \cdot d)$, and its degree is at most $2d$ as well. Applying Theorem 3.1 directly, we can certify the evaluation of P' on all n vectors of d dimensions in $\tilde{O}(n \cdot d)$ time. ◀

One consequence (among many) of Theorem A.1 is an MA-proof system for the *dominating pairs* problem in computational geometry: given a set S of n vectors in $\mathbb{R}^d$, determine if there are $u, v \in S$ such that $u[i] < v[i]$ for all $i = 1, \ldots, d$. (Here, our computational model is the real RAM, where additions and comparisons of reals are unit time operations.)

▶ **Corollary A.2.** *There is an MA-proof system for counting the number of dominating pairs in $\tilde{O}(n^{1.5} \cdot d^{1.5})$ time. As a consequence, there is a MA-proof system for counting* 0-1 *solutions to a linear program with k variables and m constraints that runs in $2^{3k/4} \cdot poly(m,k)$ time.*

Proof. Given that one can count orthogonal vectors of n vectors in d Boolean dimensions in $t(n,d)$ time, a recent reduction of Chan and the author [17] shows how to count the number of dominating pairs among n vectors in $\mathbb{R}^d$, in $O(n^2d^2/s + t(n, 2+ds))$ time, for any positive natural number s. In fact, the reduction makes precisely one call to orthogonal vectors. Theorem A.1 provides an $\tilde{O}(n \cdot d)$ time proof system for counting orthogonal vectors, so by setting $s = \sqrt{n \cdot d}$ to balance the factors, there is a proof system for counting dominating pairs in $\tilde{O}(n^{1.5} \cdot d^{1.5})$ time. By a reduction of Impagliazzo, Paturi, and Schneider [30] from integer linear programming to dominating pairs, we obtain an MA-proof system for counting

the number of Boolean solutions to a linear program with k variables and m inequalities in $2^{3k/4} \cdot \text{poly}(m, k)$ time. ◀

Finally, we illustrate that the above ideas can certify Nearest Neighbors (in the Hamming metric) in near-linear time as well:

Reminder of Theorem 1.6 Let $d \leq n$. For every $A \subseteq \{0,1\}^d$ with $|A| = n$, and every parameter $k \in \{0, 1, \ldots, d\}$, there is an MA-proof system certifying for every $v \in A$ the number of points in A with Hamming distance at most k from v, running in $\tilde{O}(n \cdot d)$ time with error probability $1/\text{poly}(n)$.

Proof. (Sketch) Analogous to Theorem A.1. Let p be a prime greater than $n^2 \cdot (2d+1)$, and let $k \in \{0, 1, \ldots, d\}$ be our proximity parameter. Define the degree-$2d$ polynomial $\Psi(x)$ to be 0 on all $j = -d, \ldots, d-2k$, and 1 on all $j = d-2k, \ldots, d$. Note that such a Ψ can easily be constructed by interpolation in $\tilde{O}(d)$ time (cf. Theorem 2.2). Define the $2d$-variable polynomial

$$P(x_1, \ldots, x_d, y_1, \ldots, y_d) := \Psi\left(\sum_{i=1}^{d} x_i \cdot y_i\right).$$

Observe that $\deg(P) \leq 2d$, and for a pair of Boolean vectors $u, v \in \{-1, 1\}^d$, $P(u, v) = 1$ if and only if u and v differ in at most k coordinates. (Differing in k coordinates is equivalent to summing $(d-k)$ ones and k minus-ones in the inner product.) Therefore, if we map all the 0/1 vectors in A to $1/-1$ vectors (mapping 0 to 1, and mapping 1 to -1), the polynomial

$$P'(u[1], \ldots, u[d]) := \sum_{j=1,\ldots,n} P(u[1], \ldots, u[d], v_j[1], \ldots, v_j[d])$$

counts the number of vectors in A (construed as vectors in $\{-1, 1\}$, instead of $\{0, 1\}$) that have Hamming distance at most k from the input $u \in \{-1, 1\}^d$. The size of P' is $O(n \cdot d)$, its degree is at most $2d$, and applying Theorem 3.1 allows us to certify the evaluation of P' on all n vectors of d dimensions in $\tilde{O}(n \cdot d)$ time. Our prime p is chosen large enough so that the values of all intermediate computations are preserved. ◀

A.1 Certifying the Number of Small Cliques

The final result of this section gives an efficient MA-proof system for verifying the number of k-cliques in a graph:

Reminder of Theorem 1.7 For every k, there is a MA-proof system such that for every graph G on n nodes, the verifier certifies the number of k-cliques in G using $\tilde{O}(n^{\lfloor k/2 \rfloor + 2})$ time, with error probability $1/\text{poly}(n)$.

Proof. The strategy (as in previous proofs) is to reduce the problem to multipoint evaluation of an appropriate circuit on an appropriate list of points, and appeal to Theorem 3.1.

Given a graph $G = (V, E)$ on n nodes with $V = [n]$, let A be its adjacency matrix. Let ℓ-Cliques(G) be the collection of all ℓ-cliques of G, represented as subsets of $[n]$ of cardinality ℓ. Given a subset $S \subseteq [n]$, let $J(S) := \{v \in (V - S) \mid (\forall u \in S)[(u, v) \in E]\}$ be the *joint neighborhood of* S. We denote the members of $J(S)$ as $\{u_{J(S),1}, \ldots, u_{J(S),|J(S)|}\} \subseteq [n]$. Consider the polynomial

$$C(x_1, \ldots, x_n) := \sum_{S \in \ell\text{-Cliques}(G)} E^{k-\ell}_{|J(S)|}(x_{u_{J(S),1}}, \ldots, x_{u_{J(S),|J(S)|}}),$$

where E_n^k is the kth elementary symmetric polynomial on n variables. Suppose $a = (a_1, \ldots, a_n) \in \{0,1\}^n$ contains exactly $k-\ell$ ones, and let $T_a \subseteq [n]$ be the set corresponding to a. Observe that $C(a_1, \ldots, a_n)$ equals the number of $S \subseteq (V - T_a)$ such that S is an ℓ-clique and every node of S has an edge to every node of T_a. Therefore, if we evaluate C on the indicator vectors for every $(k-\ell)$-clique in G, the sum of these evaluations will be the number of k-cliques in G times $\binom{n}{k-\ell}$ (every k-clique will be counted $\binom{n}{k-\ell}$ times in the summation).

Therefore, it suffices to evaluate C on the $O(\binom{n}{k-\ell})$ indicator vectors of $(k-\ell)$-cliques in G. These vectors of length n can obviously be prepared in $O(n^{k-\ell+1})$ time.

It is well-known that for every k, the kth elementary symmetric polynomial on variables $x_1, \ldots, x_n$ can be computed in $O(n^2)$ size and degree $O(n)$ (this result is often attributed to Ben-Or). To compute this polynomial, we just have to determine the coefficient of z^k in the polynomial

$$\prod_{i=1}^{n}(z - x_i),$$

which can be done by computing the coefficient of z^k in the polynomial determined by feeding the set of points $\{(x_0, \prod_{i=1}^{n}(x_0 - x_i)), (x_1, 0), \ldots, (x_n, 0)\}$ into a circuit for univariate interpolation, where x_0 is a point different from $x_1, \ldots, x_n$. Each of the joint neighborhoods $J(S)$ can easily be determined in $O(\ell \cdot n)$ time. The total degree of C is therefore $O(n)$, and its size is $O(n^2 \cdot \binom{n}{\ell})$.

Applying Theorem 3.1 directly, we can evaluate C on $O(\binom{n}{k-\ell})$ points over $\mathbb{F}_p$ with $p > n^{k-\ell}$, in time

$$\tilde{O}\left(\binom{n}{k-\ell} \cdot n + \binom{n}{\ell} \cdot n^2\right).$$

Setting $\ell = \lfloor k/2 \rfloor$ yields a running time of $\tilde{O}(n^{\lfloor k/2 \rfloor + 2})$. ◀

Toward the KRW Composition Conjecture: Cubic Formula Lower Bounds via Communication Complexity*

Irit Dinur[1] and Or Meir[2]

1 **Faculty of Computer Science and Mathematics, Weizmann Institute, Rehovot, Israel**
irit.dinur@weizmann.ac.il

2 **Department of Computer Science, Haifa University, Haifa, Israel**
ormeir@cs.haifa.ac.il

Abstract

One of the major challenges of the research in circuit complexity is proving super-polynomial lower bounds for de-Morgan formulas. Karchmer, Raz, and Wigderson [20] suggested to approach this problem by proving that formula complexity behaves "as expected" with respect to the composition of functions $f \diamond g$. They showed that this conjecture, if proved, would imply super-polynomial formula lower bounds.

The first step toward proving the KRW conjecture was made by Edmonds et al. [10], who proved an analogue of the conjecture for the composition of "universal relations". In this work, we extend the argument of [10] further to $f \diamond g$ where f is an arbitrary function and g is the parity function.

While this special case of the KRW conjecture was already proved implicitly in Håstad's work on random restrictions [14], our proof seems more likely to be generalizable to other cases of the conjecture. In particular, our proof uses an entirely different approach, based on communication complexity technique of Karchmer and Wigderson [21]. In addition, our proof gives a new structural result, which roughly says that the naive way for computing $f \diamond g$ is the *only* optimal way. Along the way, we obtain a new proof of the state-of-the-art formula lower bound of $n^{3-o(1)}$ due to [14].

1998 ACM Subject Classification F.1.3 Complexity Measures and Classes, F.2 Analysis of Algorithms and Problem Complexity

Keywords and phrases Formula lower bounds, communication complexity, Karchmer-Wigderson games, KRW composition conjecture

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.3

1 Introduction

One of the major challenges in the quest for proving lower bounds is to find an explicit function that requires formulas of super-polynomial size. Formally, (de Morgan) formulas are defined as circuits with AND, OR, and NOT gates that have fan-out 1, or in other words, their underlying graph is a tree.

* This research was partially supported by Irit Dinur's ERC grant number 239986.

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 3; pp. 3:1–3:51

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

The state-of-the-art in this direction is a lower-bound of $\tilde{\Omega}(n^3)$ due to Håstad [14][1], building on earlier work by [32, 1, 16, 27]. This result was achieved by the celebrated method of random restrictions, and in particular, by providing a lower-bound on the shrinkage exponent, which is the parameter controlling the effect of random restrictions. Håstad's lower bound on the shrinkage exponent is known to be best possible, so improving the cubic lower-bound requires a new approach.

In this work we pursue a different approach following the KRW conjecture, named after Karchmer, Raz, and Wigderson who suggested this conjecture in [20]. The KRW conjecture is about composed functions of the form $f \diamond g : \{0,1\}^{mn} \to \{0,1\}$ defined by

$$f \diamond g(x_1, \ldots, x_m) = f(g(x_1), \ldots, g(x_m)),$$

where $f : \{0,1\}^m \to \{0,1\}$ and $g : \{0,1\}^n \to \{0,1\}$. The conjecture says roughly[2] that

$$\mathsf{L}(f \diamond g) \approx \mathsf{L}(f) \cdot \mathsf{L}(g)$$

where $\mathsf{L}(\cdot)$ denotes the formula size of a function, namely, the number of leaves in the underlying tree. In other words, the conjecture says that the naive way of computing $f \diamond g$, by first computing g on each component and then f, is essentially the best way to do it. In addition to being interesting in its own right, the KRW conjecture is particularly important due to the fact that it implies super-polynomial lower bounds for an explicit function [20].

Despite some early successes in the study of the KRW conjecture [10, 15], so far it has not bore new lower bounds. Recently, Gavinsky et al. [12] have made the first progress in two decades in this direction. In this work, we push this direction further, and obtain a new proof of the state-of-the-art cubic lower bound on the formula size of Andreev's function.

▶ **Theorem 1.1.** *Let* $And_n : \{0,1\}^n \to \{0,1\}$ *be Andreev's function* [1] *over* n *bits. Then,*

$$\mathsf{L}(And_n) \geq n^{3-o(1)}.$$

Although this was already proved by [14], our proof is based on an entirely different method – specifically, the communication-complexity technique of Karchmer and Wigderson [21]. Unlike the proof by random restrictions, this method does not seem to have any inherent limitation, and we do not see a reason why it should not be able to prove stronger lower bounds. More importantly, we see this work as a step toward proving the KRW conjecture.

Toward proving the KRW conjecture

As a first step toward proving their conjecture, [20] suggested to study the composition of *universal relations*, which are objects that are similar to functions but are easier to analyze in this context. Let us denote the universal relation by U. Then, [20] suggested to prove an analogue of their conjecture for the composition $U \diamond U$. This challenge was met by Edmonds et al. [10], and an alternative proof was discovered later by Håstad and Wigderson [15].

Recently, Gavinsky et al. [12] made further progress and proved an analogue of the KRW conjecture for $f \diamond U$: the composition of an arbitrary function f with the universal relation. Thus, the next step to proving the KRW conjecture would be to replace the universal relation in their result with a function g, for every choice of g. In this work, we do it for the special case where g is the parity function over n bits, denoted $\oplus_n$.

---

[1] Recently, Tal [34] provided a new proof of the lower bound on the shrinkage exponent, and along the way improved the lower order factors in Håstad's lower bound.

[2] The original KRW conjecture was formulated in terms of formula depth, this variant with formula size is from [12].

▶ **Theorem 1.2** (Main theorem). *Let $f : \{0,1\}^m \to \{0,1\}$ be a non-constant function. Then,*

$$\mathsf{L}(f \diamond \oplus_n) \geq \frac{\mathsf{L}(f) \cdot \mathsf{L}(\oplus_n)}{2^{\tilde{O}(\sqrt{m+\log n})}}.$$

To summarize, the KRW conjecture has been verified on $U \diamond U$ [10], and then $f \diamond U$ [12]. In this work we prove it for $f \diamond \oplus_n$, and one would hope that the next step(s) would lead to $f \diamond g$ for every g.

It is important to note that lower bounds on the composition $f \diamond \oplus_n$ were already proved implicitly in the aforementioned works on the Andreev's function [1, 16, 27, 14, 34]. In particular, [14, 34] implicitly prove that

$$\mathsf{L}(f \diamond \oplus_n) \geq \frac{\mathsf{L}(f) \cdot \mathsf{L}(\oplus_n)}{\mathrm{poly}(\log m, \log n)}.$$

However, our proof seems more likely to be generalized to other choices of g, and in addition, it gives a structural inverse result: not only is the naive way to compute $f \diamond \oplus_n$ optimal in terms of complexity, but it is essentially the *only* optimal way to compute $f \diamond \oplus_n$. More specifically, we show that any formula computing $f \diamond \oplus_n$ with near optimal complexity must incur a cost of $\approx \mathsf{L}(\oplus_n)$ before starting the computation of f. We discuss this result a bit more in Section 1.2 below, and a formal description is given in Section 3.

Bypassing a barrier for Karchmer-Wigderson relations

As all the previous works on the KRW conjecture, our proof is based on a method of Karchmer and Wigderson [21]. A particularly interesting feature of our proof of Theorem 1.1 is that it is the first proof of a super-quadratic formula lower bound that uses this method. In particular, this requires bypassing a known barrier for Karchmer-Wigderson relations, see Section 1.1 below for more detail.

Average-case lower bounds

A recent line of research [30, 18, 24, 8, 34] extended the aforementioned formula lower-bounds to the average-case setting. Our proof can be extended to the average-case setting as well, yielding the following results. In what follows, we say that a function F is (s,ε)-*hard* if every formula of size at most s computes F correctly on at most $\frac{1}{2}+\varepsilon$ fraction of the inputs.

▶ **Theorem 1.3.** *Let $f : \{0,1\}^m \to \{0,1\}$ be an (s,ε)-hard function. Then, $f \diamond \oplus_n$ is $(s', \varepsilon + 2^{-m})$-hard for*

$$s' \geq s \cdot \mathsf{L}(\oplus_n)/2^{\tilde{O}(\sqrt{m+\log n})}.$$

▶ **Corollary 1.4.** *For every $n, c \in \mathbb{N}$ there exists a function $F_{n,c} : \{0,1\}^n \to \{0,1\}$ bits that is (S, n^{-c})-hard for*

$$S \geq n^{3-\tilde{O}(\frac{1}{\sqrt{\log n}})}.$$

1.1 Background: Karchmer-Wigderson relations

Karchmer and Wigderson [21] observed an interesting connection between depth complexity and communication complexity: for every boolean function f, there exists a corresponding communication problem KW_f, such that any *deterministic* protocol for solving KW_f can be

syntactically converted to a formula computing f, and vice versa. In particular, the depth complexity of f is equal to the deterministic communication complexity of KW_f and the formula size of f equals the *protocol size* of KW_f, which is the smallest number of transcripts in a deterministic protocol that solves KW_f. The communication problem KW_f is often called the *Karchmer-Wigderson relation* of f, and we will refer to it as a *KW relation* for short.

The KW relation KW_f is defined as follows: Alice gets an input $x \in f^{-1}(0)$, and Bob gets as input $y \in f^{-1}(1)$. Clearly, it holds that $x \neq y$. The goal of Alice and Bob is to find a coordinate i such that $x_i \neq y_i$. Note that there may be more than one possible choice for i, which means that KW_f is a relation rather than a function. In what follows, we denote the communication complexity and protocol size of KW_f by $\mathsf{C}(KW_f)$ and $\mathsf{L}(KW_f)$ respectively.

The randomized-complexity barrier

KW relations allow us to translate questions about formula complexity to questions about communication complexity, thus giving us a different angle for attacking those questions. This method had great success in proving *monotone* formula lower-bounds [21, 13, 28, 20], culminating in exponential formula lower-bounds [28].

In contrast, in the non-monotone setting, this method has been stuck so far at proving quadratic lower-bounds. This is no coincidence: unlike the monotone setting, in the general setting it is known that every KW relation can be solved by a *randomized* protocol of quadratic size. Therefore, we cannot hope to prove better lower bounds using techniques that work against randomized protocols, and this fact severely restricts the techniques that we may employ. In particular, as noted by [12], this barrier implies that KW relations do not have "hard distributions", i.e., distributions over the inputs that are hard for every deterministic protocol. This fact makes it difficult to analyze those relations using information-theoretic techniques, and similar reasons prohibit the use of rectangle-based techniques [19].

As mentioned above, our proof of Theorem 1.1 is the first proof of a super-quadratic lower-bound using KW relations. In particular, our proof is the first to bypass the randomized-complexity barrier.

1.2 Proof outline

In order to prove Theorem 1.2, we analyze $KW_{f \diamond g}$ (for the case of $g = \oplus_n$) and show that

$$\mathsf{C}(KW_{f \diamond g}) \approx \mathsf{C}(KW_f) + \mathsf{C}(KW_g).$$

(We actually prove a similar but stronger statement, namely $\log \mathsf{L}(KW_{f \diamond g}) \approx \log \mathsf{L}(KW_f) + \log \mathsf{L}(KW_g)$ but for this outline we shall focus on the communication complexity.)

In the KW relation $KW_{f \diamond g}$, Alice and Bob's inputs are conveniently viewed as $m \times n$ matrices X, Y, respectively, such that $g(X) \in f^{-1}(0)$ and $g(Y) \in f^{-1}(1)$, where $g(X) \in \{0,1\}^m$ is obtained by applying g to each row of X and similarly $g(Y)$. Their goal is to find an entry (i, j) such that $X_{i,j} \neq Y_{i,j}$.

The naive protocol for Alice and Bob is as follows. Alice computes $a = g(X)$ and Bob computes $b = g(Y)$. In the first stage they solve KW_f on a, b and find an index $i \in [m]$ where $a_i \neq b_i$. Then, in the second stage, then solve KW_g on inputs X_i, Y_i to find j as required. This protocol shows that $\mathsf{C}(KW_{f \diamond g}) \leq \mathsf{C}(KW_f) + \mathsf{C}(KW_g)$. We remark that the naive strategy for $KW_{f \diamond g}$ corresponds to the naive formula for $f \diamond g$, but note that the order is reversed (top-down vs. bottom up).

The KRW conjecture asserts that the naive protocol for $KW_{f\diamond g}$ is essentially optimal. A natural approach for proving the KRW conjecture is to show that any optimal protocol that solves $KW_{f\diamond g}$ must behave approximately like the naive protocol. This approach potentially gives, in addition to a lower bound, a *structural result* about optimal protocols for $KW_{f\diamond g}$. This approach was first taken in [10] for the composition of two universal relations. In this work, we extend the argument of [10] to the case where f is an arbitrary function and $g = \oplus_n$ is the parity function.

Why should it be the case the any optimal behaves like the naive protocol? In order to gain intuition, consider the following thought experiment: Suppose that every message of Alice and Bob was either only "about" $g(X)$ and $g(Y)$, or only "about" X_i and Y_i for some $i \in [m]$. Intuitively, in the first case they are trying to solve KW_f on $g(X)$ and $g(Y)$, and in the second case they are trying to solve KW_g on some pair of rows X_i and Y_i. We now claim that if such a protocol was optimal, then Alice and Bob would have had to finish solving KW_f before solving KW_g on any pair of rows, or in other words, they would have had to behave as in the naive protocol.

More specifically, we claim that it only makes sense for Alice and Bob to communicate about a pair of rows X_i and Y_i if they already know that $g(X_i) \neq g(Y_i)$. To see why this is true, suppose that Alice and Bob communicate about some X_i and Y_i without knowing whether $g(X_i) \neq g(Y_i)$ or not. In such a case, Alice and Bob might send a lot of bits about X_i and Y_i, only to find out eventually that $X_i = Y_i$. This would mean that their effort has been in vain, since if $X_i = Y_i$ then the answer to $KW_{f\diamond g}$ cannot possibly lie in X_i and Y_i. Hence, if Alice and Bob do not wish to waste bits on rows where $X_i = Y_i$, they should first make sure that $g(X_i) \neq g(Y_i)$. However, finding $i \in [m]$ such that $g(X_i) \neq g(Y_i)$ requires solving KW_f on $g(X)$ and $g(Y)$. Therefore, Alice and Bob must solve KW_f before solving KW_g. We now discuss how to turn this intuitive argument into a formal proof.

We begin with an arbitrary optimal protocol Π for $KW_{f\diamond g}$, and show that it has an approximate two-stage structure similar to the naive protocol in the following sense. We split transcripts of Π into two parts π_1 and π_2, supposedly corresponding to the stages of solving KW_f and KW_g respectively. We identify a collection of partial transcripts π_1 that did not fully solve a certain random embedding of KW_f into $KW_{f\diamond g}$. We call these partial transcripts "alive" since the proof focuses only on them and shows that they lead to many distinct leaves of the protocol. We refer the reader to Section 3 for more details, and remark that this embedding is generic and allows embedding KW_f into $KW_{f\diamond g}$ for any choice of g. We then prove:

1. **The first stage is hard:** There are live partial transcripts π_1 whose length is almost about $\mathsf{C}(KW_f)$.
2. **The second stage is hard:** If π_1 is alive, then there is some π_2 whose length is about $\mathsf{C}(KW_g)$.

These two items together imply that Π has a transcript whose length is

$$|\pi_1| + |\pi_2| \approx \mathsf{C}(KW_f) + \mathsf{C}(KW_g).$$

In addition, observe that the second item implies a structural result on the optimal protocols for $KW_{f\diamond g}$: Essentially, this item says that as long as Alice and Bob have not solved KW_f on $g(X)$ and $g(Y)$, they must still incur a cost of $\mathsf{C}(KW_g)$. This roughly means that in any optimal protocol, Alice and Bob must first solve KW_f and then solve KW_g. Translating this result from the language of KW relations to the language of formulas, this means that any optimal formula for $f \diamond g$ must first compute g and then compute f (in the case of $g = \oplus_n$).

Our definition of π_1 being alive makes it not too difficult to prove the first item above (see the f-stage lemma in Section 4). However, the second item is much more technically difficult. Here we must prove that in order to solve KW_g on one of the m rows of X, Y, Alice and Bob must communicate $\mathsf{C}(KW_g)$ bits. The difficulty is that since Alice and Bob already spoke $|\pi_1|$ bits, they are not playing on all possible input pairs X, Y but rather on a residual rectangle that depends on π_1.

Nevertheless, since $|\pi_1| \leq m$, they only communicated about one bit on the average row. Intuitively, this means that on the typical row, the players should be quite far from solving KW_g. Hence, if they try to finish solving $KW_{f \diamond g}$ on one of those typical rows, they will have to communicate about $\mathsf{C}(KW_g)$ bits. However, there can be a few "revealed" rows on which π_1 reveals a lot, and on which it might be easier to solve $KW_{f \diamond g}$. We therefore take steps to *force* Alice and Bob to play on the typical "non-revealed" rows. In order to carry out our approach two ingredients are necessary:

- The first ingredient is a way to measure how much progress the players made on a given row, in a way that guarantees there will only be a few revealed rows. Luckily, for the parity function $g = \oplus_n$, this progress is directly related to the *information* that was communicated on the row. We then use an averaging argument which implies that on most rows, π_1 reveals at most one bit of information (and hence, only one bit of progress was made).
- The second ingredient is a way to force Alice and Bob to play only on the non-revealed rows. This is done by forcing X and Y to be identical on the revealed rows (so the final output (i, j) cannot be in these rows). Formally, this is done by focusing on a sub-rectangle of the residual rectangle of π_1, in which X and Y are identical on the revealed rows. However, one must do this without losing the complexity of the problem. Showing that this is possible is highly non-trivial, and is the most difficult part of our argument. The main difficulty comes from the fact that if, in the residual rectangle of π_1, it holds that $g(X_i) \neq g(Y_i)$ for some revealed row (X_i, Y_i), then we cannot force X_i and Y_i to be identical. The point is that such a situation cannot occur, because π_1 is alive, i.e., it has not fully solved KW_f yet. This implies that π_1 has could not find a small set of (revealed) rows in which the answer to KW_f lies. Thus, Alice and Bob cannot rule out that $g(X_i) = g(Y_i)$ in any revealed row i.

In implementing the two above ingredients, we develop two new tools that might be of use to future works:

- **Averaging argument for min-entropy:** In the discussion above, we argued that Alice and Bob gained only very little information on the *average* row of X and Y and therefore, by an averaging argument, this holds for *most* rows of X and Y. Such an averaging argument is easy to prove when we model information using Shannon entropy. Edmonds et al. [10], whose argument we extend, could not use Shannon entropy in their argument. Therefore, they defined another measure of information called "predictability" and proved an averaging argument for this measure.

  For our argument, neither Shannon entropy nor predictability are appropriate, and instead we model information using min-entropy. This requires us to prove a (non-trivial) averaging argument for min-entropy – see Section 6.1 for details.
- **Fortification lemma:** Throughout our proof, we often need to connect statements about information to statements about complexity. For example, we would like to say things like "Alice and Bob learned only little information, so the complexity of solving KW_f has not decreased by much". The reason is that in the implementation of the second ingredient, we restrict ourselves to a sub-rectangle. This restriction effectively gives information to

Alice and Bob, and we need to make sure that this information does not allow them to solve KW_f prematurely.
However, information is not always related to complexity. In particular, it is possible to come up with examples for relations KW_f in which Alice and Bob may get little information while reducing the complexity by much, or vice versa. In order to resolve this issue, we prove a general "fortification lemma", which shows that every relation KW_f has a sub-relation KW'_f for which the information and the complexity are related – see Section 6.2 for details.

1.3 Organization of the paper

We cover the required preliminaries in Section 2. Then, in Section 3, we prove our main theorem (Theorem 1.2), as well as our structural result and the resulting cubic lower bounds (Theorem 1.1). The proof of the main theorem uses three lemmas, which are proved in Sections 4, 5 and 7. We develop the new tools discussed above in Section 6. We extend our main theorem and the cubic lower bounds to the average-case setting in Section 8. Finally, in Section 9, we discuss some future directions and suggest some open problems whose solution might bring us closer to proving the KRW conjecture.

2 Preliminaries

We use $[n]$ to denote the set $\{1,\ldots,n\}$. Given two strings $x,y \in \{0,1\}^n$, the relative *(Hamming)* distance between x and y is the fraction of coordinates on which they disagree. For a function $t:\mathbb{N}\to\mathbb{N}$, we denote

$$\tilde{O}(t) \stackrel{\text{def}}{=} O(t\cdot \log^{O(1)} t)$$
$$\tilde{\Omega}(t) \stackrel{\text{def}}{=} \Omega(t/\log^{O(1)} t).$$

We denote the set of $m\times n$ binary matrices by $\{0,1\}^{m\times n}$. For every binary $m\times n$ matrix X, we denote by $X_j\in\{0,1\}^n$ the j-th row of X. Throughout the paper, we denote by $\oplus_n$ the parity function over n bits.

2.1 Formulas

▶ **Definition 2.1.** A *formula* ϕ is a binary tree, whose leaves are identified with literals of the forms x_i and $\neg x_i$, and whose internal vertices are labeled as AND ($\wedge$) or OR ($\vee$) gates. The *size* of a formula is the number of its *leaves* (which is the same as the number of its wires up to a factor of 2). We note that a single input coordinate x_i can be associated with many leaves.

▶ **Definition 2.2.** A formula ϕ computes a binary function $f:\{0,1\}^n\to\{0,1\}$ in the natural way. The *formula complexity* of a boolean function $f:\{0,1\}^n\to\{0,1\}$, denoted $\mathsf{L}(f)$, is the size of the smallest formula that computes f. The *depth complexity* of f, denoted $\mathsf{D}(f)$, is the smallest depth of a formula that computes f.

The following definition generalizes the above definitions from functions to promise problems, which will be useful when we discuss Karchmer-Wigderson relations.

▶ **Definition 2.3.** Let $\mathcal{X},\mathcal{Y}\subseteq\{0,1\}^n$ be disjoint sets. We say that a formula ϕ *separates* $\mathcal{X}$ and $\mathcal{Y}$ if $\phi(\mathcal{X})=0$ and $\phi(\mathcal{Y})=1$. The *formula complexity of the rectangle* $\mathcal{X}\times\mathcal{Y}$, denoted $\mathsf{L}(\mathcal{X}\times\mathcal{Y})$, is the size of the smallest formula that separates $\mathcal{X}$ and $\mathcal{Y}$. The *depth*

complexity of the rectangle $\mathcal{X} \times \mathcal{Y}$, denoted $\mathsf{D}(\mathcal{X} \times \mathcal{Y})$, is the smallest depth of a formula that separates $\mathcal{X}$ and $\mathcal{Y}$.

Note that Definition 2.2 is indeed a special case of Definition 2.3 where $\mathcal{X} = f^{-1}(0)$ and $\mathcal{Y} = f^{-1}(1)$. The following theorem establishes a tight connection between the formula complexity and the depth complexity of a function.

▶ **Theorem 2.4** ([4], following [31, 7]). *For every $\alpha > 0$ the following holds: For every formula ϕ of size s, there exists an equivalent formula ϕ' of depth at most $O(2^{\frac{1}{\alpha}} \cdot \log s)$ and size at most $s^{1+\alpha}$.*

2.2 Communication complexity

Let $\mathcal{X}$, $\mathcal{Y}$, and $\mathcal{O}$ be sets, and let $R \subseteq \mathcal{X} \times \mathcal{Y} \times \mathcal{O}$ be a relation. The communication problem [35] that corresponds to R is the following: two players, Alice and Bob, get inputs $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, respectively. They would like to find $o \in \mathcal{O}$ such that $(x, y, o) \in R$. To this end, they send bits to each other until they find o, but they would like to send as few bits as possible. The communication complexity of R is the minimal number of bits that is transmitted by any protocol that solves R. More formally, we define a protocol as a binary tree, in which every vertex represents a possible state of the protocol, and every edge represents a message that moves the protocol from one state to another:

▶ **Definition 2.5.** A *(deterministic) protocol* that solves a relation $R \subseteq \mathcal{X} \times \mathcal{Y} \times \mathcal{O}$ is a rooted binary tree with the following structure:

- Every node of the tree is labeled by a rectangle $\mathcal{X}_v \times \mathcal{Y}_v$ where $\mathcal{X}_v \subseteq \mathcal{X}$ and $\mathcal{Y}_v \subseteq \mathcal{Y}$. The root is labeled by the rectangle $\mathcal{X} \times \mathcal{Y}$. Intuitively, the rectangle $\mathcal{X}_v \times \mathcal{Y}_v$ is the set of pairs of inputs that lead the players to the vertex v.
- Each internal vertex v is *owned* by Alice or by Bob. Intuitively, v is owned by Alice if it is Alice's turn to speak at state v, and same for Bob.
- Every edge of the tree is labeled by either 0 or 1.
- For every internal vertex v that is owned by Alice, the following holds: Let v_0 and v_1 be the children of v associated with the out-going edges labeled with 0 and 1, respectively. Then,
  - $\mathcal{X}_v = \mathcal{X}_{v_0} \cup \mathcal{X}_{v_1}$, and $\mathcal{X}_{v_0} \cap \mathcal{X}_{v_1} = \emptyset$.
  - $\mathcal{Y}_v = \mathcal{Y}_{v_0} = \mathcal{Y}_{v_1}$.

  Intuitively, when the players are at the vertex v, Alice sends 0 to Bob if her input is in $\mathcal{X}_{v_0}$ and 1 if her input is in $\mathcal{X}_{v_1}$. An analogous property holds for vertices owned by Bob, while changing the roles of $\mathcal{X}$ and $\mathcal{Y}$.
- For each leaf ℓ, there exists a value o such that $\mathcal{X}_\ell \times \mathcal{Y}_\ell \times \{o\} \subseteq R$. Intuitively, o is the output of the protocol at ℓ.

▶ **Definition 2.6.** Given a protocol Π and a vertex v of Π, the *transcript of* v is the string that is obtained by concatenating the labels of the edges on the path from the root to v. Intuitively, this string consists of the messages that Alice and Bob sent in their conversation until they got to v. Since the transcript determines v uniquely and vice versa, we will often identify the transcript with the vertex v. If v is a leaf of the protocol, we say that it is a *full transcript*, and otherwise we say that it is a *partial transcript*. Unless stated explicitly otherwise, whenever we say "transcript" we mean "full transcript".

Given a pair of inputs $(x, y) \in \mathcal{X} \times \mathcal{Y}$, we define the transcript of (x, y), denoted $\Pi(x, y)$, as the (full) transcript of the protocol when Alice and Bob get the inputs x and y respectively.

More formally, Let ℓ be the unique leaf ℓ such that $(x,y) \in \mathcal{X}_\ell \times \mathcal{Y}_\ell$, and define $\Pi(x,y)$ be the transcript of ℓ.

▶ **Definition 2.7.** The *communication complexity* of a protocol Π, denoted $\mathsf{C}(\Pi)$, is the the depth of the protocol tree. In other words, it is the maximum number of bits that can be sent in an execution of the protocol on any pair of inputs (x,y). For a relation R, we denote by $\mathsf{C}(R)$ the minimal communication complexity of a (deterministic) protocol that solves R.

▶ **Definition 2.8.** We define the *size* of a protocol Π to be its number of leaves. Note that this is also the number of distinct full transcripts of the protocol. We define the *protocol size*[3] of a relation R, denoted $\mathsf{L}(R)$, as the size of the smallest protocol that solves it.

2.3 Karchmer-Wigderson relations

In this section, we define KW relations formally, and state the correspondence between KW relations and formulas. We start by defining KW relations for general rectangles, and then specialize the definition to functions.

▶ **Definition 2.9.** Let $\mathcal{X},\mathcal{Y} \subseteq \{0,1\}^n$ be two disjoint sets. The *KW relation* $KW_{\mathcal{X}\times\mathcal{Y}} \subseteq \mathcal{X} \times \mathcal{Y} \times [n]$ is defined by

$$KW_{\mathcal{X}\times\mathcal{Y}} \stackrel{\text{def}}{=} \{(x,y,i) : x_i \neq y_i\}$$

Intuitively, $KW_{\mathcal{X}\times\mathcal{Y}}$ corresponds to the communication problem in which Alice gets $x \in \mathcal{X}$, Bob gets $y \in \mathcal{Y}$, and they would like to find a coordinate $i \in [n]$ such that $x_i \neq y_i$ (note that $x \neq y$ since $\mathcal{X} \cap \mathcal{Y} = \emptyset$).

▶ **Definition 2.10.** Let $f : \{0,1\}^n \to \{0,1\}$ be a non-constant function. The *KW relation of* f, denoted KW_f, is defined by $KW_f \stackrel{\text{def}}{=} KW_{f^{-1}(0)\times f^{-1}(1)}$.

We are now ready to state the connection between formulas and KW relations. We state the connection for general rectangles, and the specialization to functions is straightforward.

▶ **Theorem 2.11** (Implicit in [21][4])**.** *Let $\mathcal{X},\mathcal{Y} \subseteq \{0,1\}^n$ be two disjoint sets. Then, for every formula ϕ that separates $\mathcal{X}$ and $\mathcal{Y}$, there exists a protocol Π_ϕ that solves $KW_{\mathcal{X}\times\mathcal{Y}}$, whose underlying tree is the same as the underlying tree of ϕ. In the other direction, for every protocol Π that solves $KW_{\mathcal{X}\times\mathcal{Y}}$ there exists a formula ϕ_Π that separates $\mathcal{X}$ and $\mathcal{Y}$, whose underlying tree is the same as the underlying tree of Π.*

▶ **Corollary 2.12** ([21])**.** *For every two disjoints sets $\mathcal{X},\mathcal{Y} \subseteq \{0,1\}^n$ it holds that $\mathsf{D}(\mathcal{X}\times\mathcal{Y}) = \mathsf{C}(KW_{\mathcal{X}\times\mathcal{Y}})$, and $\mathsf{L}(\mathcal{X}\times\mathcal{Y}) = \mathsf{L}(KW_{\mathcal{X}\times\mathcal{Y}})$. In particular, for every non-constant $f : \{0,1\}^n \to \{0,1\}$, it holds that $\mathsf{D}(f) = \mathsf{C}(KW_f)$, and $\mathsf{L}(f) = \mathsf{L}(KW_f)$.*

Note that due to the connection between formula depth and formula size (Theorem 2.4), it holds that the communication complexity $\mathsf{C}(KW_f)$ and the logarithm of the protocol size $\log \mathsf{L}(KW_f)$ are always within constant factor of each other. In order to streamline the presentation, in many of the intuitive discussions in this paper we will identify those two measures: for example, we will say that "Alice and Bob must transmit t bits" and mean that

[3] This parameter is usually called the "protocol partition number" [25], but we prefer to use the term "protocol size" in order to streamline the presentation.

[4] This fact was discussed explicitly in [29, 19, 12].

the protocol size is at least 2^t. However, our formal results will always be about the protocol size.

Throughout this work, we will rely extensively on the following *sub-additivity property* of protocol size and formula complexity: for every $\mathcal{X}, \mathcal{Y} \subseteq \{0,1\}^n$ such that $\mathcal{X} = \mathcal{X}_0 \cup \mathcal{X}_1$ and $\mathcal{Y} = \mathcal{Y}_0 \cup \mathcal{Y}_1$, it holds that

$$\begin{aligned} \mathsf{L}(\mathcal{X} \times \mathcal{Y}) &\leq \mathsf{L}(\mathcal{X}_0 \times \mathcal{Y}) + \mathsf{L}(\mathcal{X}_1 \times \mathcal{Y}) \\ \mathsf{L}(\mathcal{X} \times \mathcal{Y}) &\leq \mathsf{L}(\mathcal{X} \times \mathcal{Y}_0) + \mathsf{L}(\mathcal{X} \times \mathcal{Y}_1). \end{aligned}$$

To see why the first inequality holds, consider the following protocol for $KW_{\mathcal{X}\times\mathcal{Y}}$: Alice starts by saying whether her input belongs to $\mathcal{X}_0$ or to $\mathcal{X}_1$. Then, the players proceed by invoking the optimal protocol for either $KW_{\mathcal{X}_0\times\mathcal{Y}}$ or $KW_{\mathcal{X}_1\times\mathcal{Y}}$. It is easy to see that the size of this protocol is at most $\mathsf{L}(\mathcal{X}_0 \times \mathcal{Y}) + \mathsf{L}(\mathcal{X}_1 \times \mathcal{Y})$. The proof of the second inequality is similar.

2.4 Information theory

We use basic concepts from information theory, see [9] for more details.

▶ **Definition 2.13** (Entropy)**.** The *entropy* of a random variable x is

$$H(x) \stackrel{\text{def}}{=} \mathbb{E}_{x_0 \leftarrow x}\left[\log \frac{1}{\Pr[x = x_0]}\right] = \sum_{x_0} \Pr[x = x_0] \cdot \log \frac{1}{\Pr[x = x_0]}.$$

The *conditional entropy* $H(x|y)$ is defined to be $\mathbb{E}_{y_0 \leftarrow y}[H(x|y = y_0)]$.

▶ **Fact 2.14.** *$H(x)$ is non-negative and is upper bounded by the logarithm of the size of the support of x. Equality is attained when x is uniformly distributed over its support.*

The notion of mutual information between two variables x and y, defined next, measures how much information x gives on y and vice versa. Intuitively, the information that x gives on y is captured by how much the uncertainty about y decreases when x becomes known.

▶ **Definition 2.15** (Mutual Information)**.** The *mutual information* between two random variables x, y, denoted $I(x : y)$ is defined as

$$I(x : y) \stackrel{\text{def}}{=} H(x) - H(x|y) = H(y) - H(y|x). \tag{1}$$

For a random variable z, the *conditional mutual information* $I(x; y|z)$ is defined as

$$I(x : y|z) \stackrel{\text{def}}{=} H(x|z) - H(x|y, z) = H(y|z) - H(y|x, z).$$

▶ **Fact 2.16.** *For all random variables x, y, z it holds that*

$$0 \leq I(x : y|z) \leq H(x|z) \leq H(x).$$

▶ **Definition 2.17.** The min-entropy of a random variable x is

$$H_\infty(x) = \min_{x_0}\left\{\log \frac{1}{\Pr[x = x_0]}\right\}.$$

In other words, $H_\infty(x)$ is the minimum number h such that $\Pr[x = x_0] = 2^{-h}$ for some x_0.

The following fact is an immediate consequence of the definitions of entropy and min-entropy.

▶ **Fact 2.18.** $H_\infty(x) \leq H(x)$.

2.5 The lower bound for parity

Since our main result is a lower bound on $KW_{f\diamond\oplus_n}$, it is helpful to recall a proof of the lower bound for $KW_{\oplus_n}$. We prove that every protocol that solves $KW_{\oplus_n}$ must transmit at least $2\log n$ bits, and more generally, must have at least n^2 distinct transcripts. We use the following fact from the field of interactive information complexity, which intuitively says that the information that Alice and Bob learn from the execution of a protocol is at most the information that an external observer learns.

▶ **Fact 2.19** ([6]). *Let Π be a protocol, and let $x$and y be random inputs to Alice and Bob in Π respectively. Let $\pi = \Pi(x,y)$ denote the transcript of Π when given x and y as inputs. Then*

$$I(\pi : x, y) \geq I(\pi : x|y) + I(\pi : y|x).$$

We also use the following definition of an edge of the boolean hypercube.

▶ **Definition 2.20.** An *edge (of the boolean hypercube)* is a pair of strings (x, y) in $\{0,1\}^n$ such that the parity of x is 0, and such that x and y differ on exactly one coordinate, which is called the *axis* of the edge.

We are now ready to prove the lower bound. The following proof is due to [12], and is based on the proof of [21].

▶ **Theorem 2.21** ([22]). *It holds that $\mathsf{L}(KW_{\oplus_n}) \geq n^2$.*

Proof. Fix a protocol Π that solves the $KW_{\oplus_n}$. Let (x, y) be a uniformly distributed edge of the hypercube, and let j denote its axis. The intuition for the proof is the following: At the end of the protocol, Alice and Bob must learn j, since it is the only valid output for (x, y). On the other hand, at the beginning of the protocol, Alice and Bob know nothing about j. Hence, throughout the protocol, each of them has to learn at least $\log n$ bits. In particular, this means that each of them has to send at least $\log n$ bits to the other, and therefore the protocol must send at least $2\log n$ bits in total.

Let $\pi = \Pi(x, y)$ be the transcript of the protocol when Alice and Bob get x and y as inputs. Since the entropy of a random variable is upper bounded by the logarithm of the size of its support, it holds that

$$\log \mathsf{L}(\Pi) \geq H(\pi) \geq I(\pi : x, y).$$

Hence, it suffices to prove that $I(\pi : x, y) \geq 2\log n$. By Fact 2.19, it holds that

$$I(\pi : x, y) \geq I(\pi : x|y) + I(\pi : y|x).$$

We prove that both terms on the right hand side the are equal to $\log n$, and this will imply the desired lower bound. For $I(\pi : y|x)$, observe that

$$I(\pi : y|x) = H(y|x) - H(y|x, \pi) = H(j|x) - H(j|x, \pi),$$

where the second equality holds because x and j together determine y, and x and y together determine j. Now, the term $H(j|x, \pi)$ is 0, because the transcript π reveals j (since it tells where x and y differ). As for the term $H(j|x)$, observe that j is uniformly distributed even conditioned on x, and therefore $H(j|x) = \log n$. It thus follows that $I(\pi : y|x) \geq \log n$. Similarly, it holds that $I(\pi : x|y) = \log n$. Those two equalities imply together than

$$\log \mathsf{L}(\Pi) \geq I(\pi : x, y) \geq I(\pi : x|y) + I(\pi : y|x) = 2\log n,$$

as required. ◀

2.6 Error-Correcting Codes

A *code* $C : \{0,1\}^n \to \{0,1\}^{n'}$ is an injective function. The images of the code C are called *codewords*, and we say that C has *relative distance* δ if the relative distance between every two distinct codewords c, c' is at least δ. The parameters n and n' are called the *message length* and the *block length* respectively. We use the following fact from coding theory:

▶ **Fact 2.22.** *Let $m, n \in \mathbb{N}$ be numbers such that $2^{m/2} \geq n$. Then, there exists a code $C : \{0,1\}^n \to \{0,1\}^{2^m}$ with relative distance at least $\frac{1}{2} - \frac{1}{2} \cdot \frac{n}{2^{m/2}}$. Furthermore, there exists a polynomial-time algorithm when given as input m, n, and $x \in \{0,1\}^n$, computes $C(x)$.*

Proof sketch. The code C is the concatenation of a Reed-Solomon code of block length $2^{m/2}$ and degree $n/(m/2)$ over $\mathbf{GF}(2^{m/2})$, and the Hadamard code of message length $m/2$. It is easy to see that the concatenated code has the required message length and block length. For the relative distance, observe that the Reed-Solomon code has relative distance $1 - \frac{n/(m/2)}{2^{m/2}} \geq 1 - \frac{n}{2^{m/2}}$, and that the Hadamard code has relative distance $\frac{1}{2}$. Hence, the concatenated code has relative distance at least $\frac{1}{2} - \frac{1}{2} \cdot \frac{n}{2^{m/2}}$, as required. ◀

We say that a code $C : \{0,1\}^n \to \{0,1\}^{n'}$ is (ρ, L)-*list decodable* if for every $w \in \{0,1\}^{n'}$, there are at most L codewords c whose relative distance to w is less than ρ. We use the following binary version of the Johnson bound, taken from [33].

▶ **Theorem 2.23** (Johnson bound). *A code $C : \{0,1\}^n \to \{0,1\}^{n'}$ with relative distance δ is (ρ, n')-list decodable for $\rho \stackrel{\text{def}}{=} \frac{1}{2} \cdot \left(1 - \sqrt{1 - 2 \cdot \delta}\right)$.*

By combining the Johnson bound with Fact 2.22, we get the following result.

▶ **Corollary 2.24.** *The code C of Fact 2.22 is $(\rho, 2^m)$-list decodable for $\rho \stackrel{\text{def}}{=} \frac{1}{2} - \frac{1}{2} \cdot \sqrt{\frac{n}{2^{m/2}}}$.*

3 Main theorem

In this section, we describe the proof of our main theorem, restated next.

▶ **Theorem 1.2** (restated). *Let $f : \{0,1\}^m \to \{0,1\}$ be a non-constant function. Then,*

$$\mathsf{L}(f \diamond \oplus_n) \geq \frac{\mathsf{L}(f) \cdot \mathsf{L}(\oplus_n)}{2^{\tilde{O}(\sqrt{m + \log n})}}.$$

We actually prove the equivalent statement that says that any protocol that solves $KW_{f \diamond \oplus_n}$ has at least $\mathsf{L}(f) \cdot \mathsf{L}(\oplus_n)/2^{\tilde{O}(\sqrt{m+\log n})}$ distinct transcripts.

The rest of this section is organized as follows: In Section 3.1, we state a structural result about protocols that solve $KW_{f \diamond \oplus_n}$. Then, in Section 3.2, we prove the structural result based on two lemmas that are proved in Sections 5 and 7 respectively. Next, in Section 3.3, we explain how to derive the main theorem from the structural result. Finally, in Section 3.4, we show how to derive the cubic lower bounds for Andreev's function from our main theorem.

3.1 The structural result

Let us recall how the communication problem $KW_{f \diamond \oplus_n}$ is defined. Alice and Bob get $m \times n$ boolean matrices X and Y and should find an entry (i, j) on which the matrices differ. Let $a, b \in \{0,1\}^m$ be the strings obtained by computing the parity of each row of X and Y respectively. Alice and Bob are guaranteed that $a \in f^{-1}(0)$ and $b \in f^{-1}(1)$. We would like

to prove that Alice and Bob must first solve KW_f on a and b, thus finding a row $i \in [m]$ such that $a_i \neq b_i$, and then solve $KW_{\oplus_n}$ on X_i and Y_i.

Fix a protocol Π for $KW_{f\diamond\oplus_n}$ and a partial transcript π_1 of Π. Intuitively, our structural result says that if Alice and Bob have not solved KW_f yet in π_1, then they have to send about $\mathsf{C}(KW_{\oplus_n})$ more bits before they finish solving $KW_{f\diamond\oplus_n}$ (actually, we will show the analogous lower bound on protocol size).

To make sense of the statement "Alice and Bob have not solved KW_f in π_1" we must first see how any protocol for $KW_{f\diamond\oplus_n}$ contains (many copies of) a protocol for KW_f. To this end, we define some notation, starting by recalling the definition of an edge.

▶ **Definition 2.20** (restated). An *edge (of the boolean hypercube)* is a pair of strings (z^0, z^1) in $\{0,1\}^n$ such that the parity of z^0 is 0 and such that z^0 and z^1 differ on exactly one coordinate, which is called the *axis* of the edge.

As we have seen in Section 2.5, the uniform distribution over edges of the boolean hypercube is a hard distribution for $KW_{\oplus_n}$. Therefore, we would like to use edges as inputs to $KW_{f\diamond\oplus_n}$. Now, an input to $KW_{f\diamond\oplus_n}$ contains m inputs to $KW_{\oplus_n}$, and this motivates the following definition.

▶ **Definition 3.1.** A *product of edges* is a pair of $m \times n$ boolean matrices $Z = (Z^0, Z^1)$ such that for every $i \in [m]$, the pair (Z_i^0, Z_i^1) is an edge. Let $\mathcal{Z} = \left\{(Z^0, Z^1)\right\}$ denote the set of all products of edges.

▶ **Definition 3.2.** Given $Z = (Z^0, Z^1) \in \mathcal{Z}$ and a string $w \in \{0,1\}^m$, we denote by Z^w the matrix defined by

$$Z_i^w \stackrel{\text{def}}{=} Z_i^{w_i}$$

for every $i \in [m]$.

Observe that for every $Z \in \mathcal{Z}$, there is a natural reduction from KW_f to $KW_{f\diamond\oplus_n}$: Given inputs $a \in f^{-1}(0)$ and $b \in f^{-1}(1)$ for Alice and Bob in KW_f, we define inputs for Alice and Bob in $KW_{f\diamond\oplus_n}$ by $X = Z^a$ and $Y = Z^b$. We now execute the protocol for $KW_{f\diamond\oplus_n}$ on X and Y, and it outputs an entry (i,j) such that $X_{i,j} \neq Y_{i,j}$. By the definition of X and Y, it follows that $a_i \neq b_i$, and therefore we obtained a solution for KW_f on a and b. The above reduction formalizes the idea that $KW_{f\diamond\oplus_n}$ contains a copy of KW_f for each Z.

Recall that we say that π_1 is *alive* if Alice and Bob have not solved KW_f in π_1. Intuitively, we will define the notion that "π_1 is alive" as follows. First, we will say π_1 is alive *with respect to a specific* Z if after the players sent π_1, they still have to send $\sqrt{m} \cdot \operatorname{poly}\log m$ bits in order to solve the copy of KW_f in $KW_{f\diamond\oplus_n}$that corresponds to Z. We will say that π_1 is alive if it is alive for many (at least 2^{-2m} fraction) of the Zs.

In order to formalize this intuitive definition, we generalize the reduction of KW_f to $KW_{f\diamond\oplus_n}$ to sub-relations of $KW_{f\diamond\oplus_n}$. Let $\mathcal{X} \subseteq (f\diamond\oplus_n)^{-1}(0)$ and $\mathcal{Y} \subseteq (f\diamond\oplus_n)^{-1}(1)$, and note that the rectangle $\mathcal{X}\times\mathcal{Y}$ defines a sub-relation $KW_{\mathcal{X}\times\mathcal{Y}}$ of $KW_{f\diamond\oplus_n}$. Now, given $Z = (Z^0, Z^1)$, we can define a corresponding sub-relation of KW_f by considering the rectangle $\mathcal{A}\times\mathcal{B}$ defined as follows:

$$\begin{aligned}\mathcal{A} &\stackrel{\text{def}}{=} \left\{a \in f^{-1}(0) | Z^a \in \mathcal{X}\right\}\\ B &\stackrel{\text{def}}{=} \left\{b \in f^{-1}(1) | Z^b \in \mathcal{Y}\right\}.\end{aligned}$$

We say that $\mathcal{A}\times\mathcal{B}$ is *the f-rectangle of $\mathcal{X}\times\mathcal{Y}$ with respect to Z*.

We are now ready to formalize what it means for a partial transcript π_1 of Π to be alive. Recall that the transcript π_1 is associated with a sub-rectangle $\mathcal{X}_{\pi_1} \times \mathcal{Y}_{\pi_1}$ of $KW_{f \diamond \oplus_n}$ in a natural way – $\mathcal{X}_{\pi_1} \times \mathcal{Y}_{\pi_1}$ contains all the pairs on inputs on which Alice and Bob transmit π_1. For every product of edges Z, we denote by $\mathcal{A}_{\pi_1,Z} \times \mathcal{B}_{\pi_1,Z}$ the f-rectangle of $\mathcal{X}_{\pi_1} \times \mathcal{Y}_{\pi_1}$ with respect to Z.

▶ **Definition 3.3.** Given a partial transcript π_1 of Π and $Z \in \mathcal{Z}$, we say that π_1 is ℓ-*alive with respect to* Z if $\mathsf{L}(\mathcal{A}_{\pi_1,Z} \times \mathcal{B}_{\pi_1,Z}) \geq 2^{\ell}$. We say that π_1 is (ℓ, α)-*alive* if it is ℓ-alive with respect to α fraction of the Z's, i.e., if

$$\Pr_{Z \in \mathcal{Z}} \left[\mathsf{L}(\mathcal{A}_{\pi_1,Z} \times \mathcal{B}_{\pi_1,Z}) \geq 2^{\ell} \right] \geq \alpha.$$

For our proof we will use $\alpha \stackrel{\text{def}}{=} 2^{-2m}$ and $\ell \stackrel{\text{def}}{=} C \cdot \sqrt{m} \cdot \log^C m$ for large enough constant $C > 0$. We say that π_1 is *alive* as short-hand for $(\ell, 2^{-2m})$-alive. We can finally state our structural result formally.

▶ **Theorem 3.4** (Structure theorem). *Let Π be a protocol for $KW_{f \diamond \oplus_n}$ and let π_1 be a live partial transcript of Π. Then, there exist at least $\mathsf{L}(\oplus_n)/2^{\tilde{O}(\sqrt{m})}$ distinct suffixes π_2 such that $\pi_1 \circ \pi_2$ is a (full) transcript of Π.*

3.2 Proof of the structure theorem

Let Π be a protocol for $KW_{f \diamond \oplus_n}$, and let π_1 be a live partial transcript of Π. Intuitively, we wish to prove that after Alice and Bob have transmitted the messages in π_1, they have to transmit another $\log \mathsf{L}(\oplus_n) - \tilde{O}(\sqrt{m})$ bits in order to solve $KW_{f \diamond \oplus_n}$. To this end, we will design a distribution over inputs $X \in \mathcal{X}_{\pi_1}$ and $Y \in \mathcal{Y}_{\pi_1}$ for Alice and Bob, and show that in order to solve $KW_{f \diamond \oplus_n}$ on inputs coming from this distribution, Alice and Bob must transmit $\log \mathsf{L}(\oplus_n) - \tilde{O}(\sqrt{m})$ bits (and thus must have $\mathsf{L}(KW_{\oplus_n})/2^{\tilde{O}(\sqrt{m})}$ distinct transcripts).

In order to design the latter distribution, we use the fact that the hardest distribution over inputs for $KW_{\oplus_n}$ is the uniform distribution over edges of the boolean hypercube (see Section 2.5). Our distribution for $KW_{f \diamond \oplus_n}$ will look roughly as follows. Let $\mathcal{Z}_{\pi_1} \subseteq \mathcal{Z}$ be the set of Zs that are "alive for π_1", i.e. for which $\mathsf{L}(\mathcal{A}_{\pi_1,Z} \times \mathcal{B}_{\pi_1,Z}) \geq 2^{\ell}$. We will choose a random $Z \in \mathcal{Z}_{\pi_1}$, then pick $a \in \mathcal{A}_{\pi_1,Z}$ and $b \in \mathcal{B}_{\pi_1,Z}$ at random, and then set $X = Z^a$ and $Y = Z^b$.

Observe that X and Y have the following property: for every $i \in [m]$, it either holds that $X_i = Y_i$ (when $a_i = b_i$) or that X_i and Y_i form an edge (when $a_i \neq b_i$). In particular, it is intuitively clear that when given inputs from this distribution, Alice and Bob must solve $KW_{\oplus_n}$ on some X_i and Y_i that form an edge. If we could show that this edge is always uniformly distributed, we could easily complete the argument by showing that Alice and Bob must send $\log \mathsf{L}(\oplus_n)$ bits. Indeed this would work if Z was uniform over *all* products of edges, i.e. if $\mathcal{Z}_{\pi_1} = \mathcal{Z}$.

Unfortunately, $\mathcal{Z}_{\pi_1}$ consists only of $\alpha = 2^{-2m}$ fraction of the products of edges, and therefore we cannot guarantee that X_i and Y_i form a *uniformly distributed* edge. However, intuitively, Alice and Bob "know" only $2m$ bits of information on Z, and therefore they know only two bits of information on the average row (X_i, Y_i). By an averaging argument, on most rows, Alice and Bob know very little information. Such rows are still hard for $KW_{\oplus_n}$, and therefore Alice and Bob must still transmit about $\log \mathsf{L}(\oplus_n)$ in order to solve $KW_{f \diamond \oplus_n}$ on one of those rows. If this is the case, we are done.

One must be careful because there could still be a few "revealed" rows on which Alice and Bob have a lot of information, and such rows might be easy for $KW_{\oplus_n}$. In order to prevent

Alice and Bob from solving $KW_{f\diamond\oplus_n}$ on those rows, we choose the distribution such that for every such row i it holds that $a_i = b_i$. This forces the equality $X_i = Y_i$, and therefore prevents Alice and Bob from solving $KW_{f\diamond\oplus_n}$ on the i-th row.

The following definition captures the essential properties of our distribution. The amount of information that Alice and Bob know about an edge is modeled using the min-entropy of the axis of the edge. The parameter t specifies the maximal amount of information that Alice and Bob may know on the axis of a row, and the set $\mathcal{R}$ consists of the rows on which Alice and Bob have too much information.

▶ **Definition 3.5.** Let X and Y be random $m \times n$ matrices. We say that (X, Y) is a *t-almost hard distribution* if there exists a set $\mathcal{R} \subseteq [m]$ such that the following properties hold:

- For every $i \in \mathcal{R}$, it holds that $X_i = Y_i$.
- For every $i \in [m] - \mathcal{R}$, there is a random coordinate j_i such that
  - Either $X_i = Y_i$, or $X_i = Y_i + \mathbf{e}_{j_i}$ i.e. X_i and Y_i form an edge with axis j_i.
  - For every specific choice X^* of X it holds that $H_\infty(j_i | X = X^*) \geq \log n - t$.
  - For every specific choice Y^* of Y it holds that $H_\infty(j_i | Y = Y^*) \geq \log n - t$.

The above argument is implemented in the following two lemmas, which are proved in Sections 7 and 5 respectively, and which together imply the structure theorem immediately. The first lemma says that there exists an almost-hard distribution over inputs that are consistent with π_1. Since this is the most involved part in our proof, we refer to it as the "main lemma".

▶ **Lemma 3.6** (Main Lemma). *Let Π be a protocol for $KW_{f\diamond\oplus_n}$, and let π_1 be a live partial transcript of Π. Then, there exists a t-almost hard distribution that is supported on $\mathcal{X}_{\pi_1} \times \mathcal{Y}_{\pi_1}$, where $t \stackrel{\text{def}}{=} c \cdot \sqrt{m} \cdot \log^c m$ for some absolute constant $c > 0$.*

The second lemma states that a hard distribution is indeed hard, i.e., that on inputs from this distribution, the players must transmit about $\log \mathsf{L}(\oplus_n)$ bits. We refer to this lemma as the "parity-stage lemma", since it analyzes the stage of the protocol Π in which the players solve $KW_{\oplus_n}$.

▶ **Lemma 3.7** (Parity-stage Lemma). *Let Π_2 be a protocol that solves $KW_{f\diamond\oplus_n}$ on a t-almost hard distribution (X, Y) with probability 1. Then, Π_2 has at least $\mathsf{L}(\oplus_n)/2^{2t}$ transcripts.*

3.3 Proof of the Main Theorem

We now explain how to prove our main theorem using the structure theorem. To this end, we use the following lemma, which says that there are many appropriate live partial transcripts π_1 to which the structure theorem can be applied. We refer to this lemma as the "f-stage lemma" since we view π_1 as the stage of the protocol in which the players solve KW_f.

▶ **Lemma 3.8** (f-stage lemma). *Let Π be a protocol for $KW_{f\diamond\oplus_n}$ of depth d. Then, there exist at least $\mathsf{L}(f)/\left(2^{\tilde{O}(\sqrt{m})} \cdot d^2\right)$ alive partial transcripts π_1 of Π, none of them is an ancestor of another.*

The intuition for the f-stage lemma is straightforward: if the players spoke less than

$$\log \mathsf{L}(f) - \tilde{O}(\sqrt{m}) \leq \mathsf{C}(KW_f) - \tilde{O}(\sqrt{m})$$

bits, then they could not have solved KW_f yet. The proof is is provided in Section 4.

We turn to the proof of the main theorem. Let Π be a protocol that solves $KW_{f\diamond\oplus_n}$. We would like to show that it has at least $\mathsf{L}(f)\cdot\mathsf{L}(\oplus_n)/2^{\tilde{O}(\sqrt{m+\log n})}$ distinct transcripts. The natural way to do so would be the following: first, we would apply the f-stage lemma to show that there are $\approx \mathsf{L}(f)$ alive partial transcripts π_1. Then, we would apply the structure theorem to those transcripts, thus showing that each of them has $\approx \mathsf{L}(g)$ suffixes. We would conclude that Π has $\approx \mathsf{L}(f)\cdot\mathsf{L}(g)$ distinct transcripts, as required.

This proof almost works, but has one issue: the f-stage lemma loses a factor that depends on the depth of Π. Thus, if Π has very large depth, the number of alive partial transcripts π_1 may be insufficient to prove the desired lower bound. In order to resolve this issue, we use a theorem that says that any protocol can be "balanced", i.e., every protocol can be transformed into an equivalent protocol whose depth is logarithmic in its size. We apply this theorem to Π to obtain a new balanced protocol Π', and then apply the foregoing proof to Π'. Specifically, we use the following theorem, which was stated in Section 2 for formulas, and which we now restate for protocols solving KW relations:

▶ **Theorem 2.4** (restated – [4], following [31, 7]). *For every $\alpha > 0$ the following holds: Let Π be a protocol of size s that solves a KW relation KW_f. Then, there exists a protocol Π' of depth at most $O(2^{\frac{1}{\alpha}}\cdot\log s)$ and size at most $s^{1+\alpha}$ that solves KW_f.*

Proof of the main theorem. Let Π be a protocol that solves $KW_{f\diamond\oplus_n}$, and let us denote its size by S. We wish to prove that $S \geq \mathsf{L}(f)\cdot\mathsf{L}(\oplus_n)/2^{\tilde{O}(\sqrt{m+\log n})}$. We may assume without loss of generality that

$$S \leq \mathsf{L}(f)\cdot\mathsf{L}(\oplus_n) \leq 2^m\cdot n^2,$$

since otherwise we are done. We apply Theorem 2.4 to Π with $\alpha = \frac{1}{\sqrt{m+\log n}}$, thus obtaining a new protocol Π' whose depth and size are

$$\begin{aligned} d' &\leq O\left(2^{\sqrt{m+\log n}}\cdot(m+2\log n)\right) = 2^{O(\sqrt{m+\log n})} \\ S' &\leq S^{1+\frac{1}{\sqrt{m+\log n}}}, \end{aligned}$$

respectively. We will prove that $S' \geq \mathsf{L}(f)\cdot\mathsf{L}(\oplus_n)/2^{\tilde{O}(\sqrt{m+\log n})}$, and this will imply the same lower bound for S as follows:

$$\begin{aligned} S &\geq (S')^{1/(1+\frac{1}{\sqrt{m+\log n}})} \\ &\geq (S')^{1-\frac{1}{\sqrt{m+\log n}}} \\ &= S'/(S')^{\frac{1}{\sqrt{m+\log n}}} \\ (\text{Since } S' \leq S^2) \quad &\geq S'/S^{\frac{2}{\sqrt{m+\log n}}} \\ (\text{Since } S \leq 2^m\cdot n^2) \quad &\geq S'/\left(2^m\cdot n^2\right)^{\frac{2}{\sqrt{m+\log n}}} \\ &= S'/2^{O(\sqrt{m+\log n})} \\ &\geq \mathsf{L}(f)\cdot\mathsf{L}(\oplus_n)/2^{\tilde{O}(\sqrt{m+\log n})}. \end{aligned}$$

In order to prove that $S' \geq \mathsf{L}(f)\cdot\mathsf{L}(\oplus_n)/2^{\tilde{O}(\sqrt{m+\log n})}$, we apply the f-stage lemma to Π', thus obtaining a collection of $\mathsf{L}(f)/2^{\tilde{O}(\sqrt{m+\log n})}$ alive partial transcripts π_1, none of which is an ancestor of another. For each such π_1, we apply the structure theorem and obtain $\mathsf{L}(\oplus_n)/2^{\tilde{O}(\sqrt{m})}$ distinct suffixes π_2 such that $\pi_1\circ\pi_2$ is a transcript of Π'. Since none of the

π_1's is an ancestor of another, all the transcripts $\pi_1 \circ \pi_2$ obtained in this way are distinct. It follows that the number of distinct transcripts $\pi_1 \circ \pi_2$ constructed in this way is at least

$$\mathsf{L}(f) \cdot \mathsf{L}(\oplus_n)/2^{\tilde{O}(\sqrt{m+\log n})},$$

as required. ◀

3.4 Cubic Lower Bounds for Andreev's Function

In this section, we derive the cubic lower bounds for Andreev's function from our main theorem. The following argument is due to Andreev [1], and was used in all the works on Andreev's function.

▶ **Theorem 1.1** (restated). *Let $And_n : \{0,1\}^n \to \{0,1\}$ be Andreev's function [1] over n bits. Then,*

$$\mathsf{L}(And_n) \geq n^{3-o(1)}.$$

Andreev's function is defined as follows: the input consists of two parts, each of length $n/2$. The first part is the truth table of a function $f : \{0,1\}^m \to \{0,1\}$ over $m \stackrel{\text{def}}{=} \log(n/2)$ bits. The second part is a sequence $x_1, \ldots, x_m$ of strings in $\{0,1\}^{n/2m}$. Andreev's function is now defined by

$$And_N(f, x_1, \ldots, x_m) \stackrel{\text{def}}{=} (f \diamond \oplus_{\frac{n}{2m}})(x_1, \ldots, x_m).$$

Proof of Theorem 1.1. It is well known that there are functions over m bits whose formula complexity is at least $2^m/\log m$ (see, e.g., [17, Theorem 1.23]). We fix the input $f : \{0,1\}^m \to \{0,1\}$ of And_n to be such a function. Clearly, the formula complexity of And_n can only be decreased by such a fixing. After the fixing, the function And_n is exactly the function $f \diamond \oplus_{\frac{n}{2m}}$. By our main theorem, the formula complexity of the latter function is at least

$$2^{m-\tilde{O}(\sqrt{m+\log n})} \cdot \left(\frac{n}{2m}\right)^2 = n^{3-\tilde{O}(\sqrt{\log n})}.$$

Therefore, the formula complexity of And_n is at least $n^{3-o(1)}$, as required. ◀

4 The f-Stage Lemma

In this section we prove the f-stage lemma. Before we restate the lemma, let us restate the definition of a *alive* partial transcript.

▶ **Definition 3.3** (restated). Given a partial transcript π_1 of Π and $Z \in \mathcal{Z}$, we say that π_1 is *ℓ-alive with respect to Z* if $\mathsf{L}(\mathcal{A}_{\pi_1,Z} \times \mathcal{B}_{\pi_1,Z}) \geq 2^\ell$. We say that π_1 is *(ℓ,α)-alive* if it is ℓ-alive with respect to α fraction of the Z's, i.e., if

$$\Pr_{Z \in \mathcal{Z}}\left[\mathsf{L}(\mathcal{A}_{\pi_1,Z} \times \mathcal{B}_{\pi_1,Z}) \geq 2^\ell\right] \geq \alpha.$$

We say that π_1 is *alive* if it is $(\ell = C \cdot \sqrt{m} \cdot \log^C m, \alpha = 2^{-2m})$-alive (where C is some large constant to be fixed later).

▶ **Lemma 3.8** (restated – f-stage lemma). *Let Π be a protocol for $KW_{f \diamond \oplus_n}$ of depth d. Then, there exist at least $\mathsf{L}(f)/\left(2^{\tilde{O}(\sqrt{m})} \cdot d^2\right)$ alive partial transcripts π_1 of Π, none of them is an ancestor of another.*

For the rest of this section, we fix Π to be a protocol for $KW_{f\diamond\oplus_n}$. Let $\ell \stackrel{\text{def}}{=} C\cdot\sqrt{m}\cdot\log^C m$ be the parameter from the definition of "alive". We will prove that Π has at least $\mathsf{L}(f)/O(2^\ell\cdot d^2)$ partial transcripts π_1 are alive, none of them is an ancestor of another.

This section is organized as follows: We start with a motivating discussion for the proof in Section 4.1. Next, in Section 4.2, we prove the f-stage lemma based on a combinatorial lemma, which is then proved in Section 4.3. Finally, in Section 4.4, we state and prove a generalization of the f-stage lemma, which will be used in Section 8 below to prove the average-case version of the main theorem.

4.1 Motivation

The basic intuition for the f-stage lemma is the following: Recall that for every product of edges $Z \in \mathcal{Z}$, there is copy of KW_f that is embedded in $KW_{f\diamond\oplus_n}$, obtained by mapping inputs a and b for KW_f into the inputs $X \stackrel{\text{def}}{=} Z^a$ and $Y \stackrel{\text{def}}{=} Z^b$ for $KW_{f\diamond\oplus_n}$.

Now, suppose that we choose a uniformly distributed $Z \in \mathcal{Z}$ and some inputs a and b according to some (unspecified) distribution, and then we run the protocol Π on inputs $X \stackrel{\text{def}}{=} Z^a$ and $Y \stackrel{\text{def}}{=} Z^b$ until it transmits $\log\mathsf{L}(f)-\ell$ bits. Let π_1 be the resulting transcript. Intuitively, since Π only transmitted $\log\mathsf{L}(f)-\ell$ bits in π_1, the players must still transmit at least ℓ bits in order to solve KW_f. On the other hand, since $\log\mathsf{L}(f)-\ell \le 2m$, the protocol has revealed at most $2m$ bits of information on Z. Therefore, we expect that after transmitting π_1, the players will still be "ℓ bits far" from solving the copy KW_f for at least 2^{-2m} fraction of the Z's – and this is roughly the definition of π_1 being alive.

The above intuitive argument can be formalized, and it shows that there exists at least one alive transcript π_1 of length $\log\mathsf{L}(f)-\ell$. However, we want to prove something stronger: we want to prove that there exist *many* alive transcripts π_1 – specifically, we wish to prove that there are about $\mathsf{L}(f)/2^\ell$ such transcripts. It turns out that this claim is more difficult to prove. To see why, it is useful to consider the following simpler version of the f-stage lemma, which refers to KW_f rather than $KW_{f\diamond\oplus_n}$:

▶ **Lemma.** *Let Π_f be a protocol that solves KW_f. Then, there exist $\mathsf{L}(f)/2^\ell$ partial transcripts π_f of Π_f whose corresponding rectangle $\mathcal{A}_{\pi_f}\times\mathcal{B}_{\pi_f}$ satisfies $\mathsf{L}(\mathcal{A}_{\pi_f}\times\mathcal{B}_{\pi_f})\ge 2^\ell$, none of them is an ancestor of another.*

It turns out that this "lemma" is false. To see why, consider the protocol Π_f for KW_f in which Alice sends Bob the unary representation of her input a – in other words, Alice views a as a number and sends the string 1^a0 to Bob. After receiving Alice's message, Bob knows a coordinate i such that $a_i \ne b_i$ and sends it to Alice using $\log m$ bits. It is now easy to see that every partial transcript π of the form 1^t0 satisfies

$$\mathsf{L}(\mathcal{A}_\pi\times\mathcal{B}_\pi)\le m \ll 2^\ell.$$

Therefore, the only partial transcripts π_f for which $\mathsf{L}(\mathcal{A}_{\pi_f}\times\mathcal{B}_{\pi_f})\ge 2^\ell$ are those of the form 1^t for some $t\in\mathbb{N}$. However, it is obvious that we cannot find even two such transcripts such that neither of them is an ancestor of the other, and therefore the claim is false.

A notable feature of the counterexample Π_f above is that it is very unbalanced; in particular, its depth is more than 2^m. It turns out that a variant of the above lemma holds if we consider only protocols Π_f that are not too deep. Specifically, we have the following result.

▶ **Lemma 4.1.** *Let Π_f be a protocol of depth d that solves KW_f. Then, there exist*

$$\frac{\mathsf{L}(f)}{(d+1)\cdot d\cdot 2^{\ell}}$$

partial transcripts π_f of Π_f whose corresponding rectangle $\mathcal{A}_{\pi_f}\times\mathcal{B}_{\pi_f}$ satisfies $\mathsf{L}(\mathcal{A}_{\pi_f}\times\mathcal{B}_{\pi_f})\geq 2^{\ell}$, none of them is an ancestor of another.

As far as we know, Lemma 4.1 is new, and we believe that it is interesting in its own right. In order to go from Lemma 4.1 to the f-stage lemma, we first observe that the only property of protocols that Lemma 4.1 uses is the fact that the protocol size $\mathsf{L}(\cdot)$ is a sub-additive measure. We therefore generalize Lemma 4.1 to a general lemma about sub-additive measures on trees:

▶ **Definition 4.2.** Given a rooted binary tree $T=(V,E)$, we say that $\phi : V\to\mathbb{N}$ is a *sub-additive measure on T* if for every vertex u with children v and w in T it holds that $\phi(u)\leq\phi(v)+\phi(w)$.

▶ **Lemma 4.3.** *Let $T=(V,E)$ be a rooted binary tree with root r and depth d, and let ϕ be a sub-additive measure on T. Suppose that there is some $t_0\in\mathbb{N}$ such that $\phi(l)\leq t_0$ for every leaf l of T. Then, for every $t\in\mathbb{N}$ such that $t\geq t_0$ there are at least*

$$\left\lfloor\frac{\phi(r)}{(d+1)\cdot d\cdot t}\right\rfloor$$

vertices v with $\phi(v)\geq t$, none of which is the ancestor of another.

Lemma 4.1 is a special case of Lemma 4.3 where the tree T is the protocol Π_f, and where the sub-additive measure ϕ is defined by

$$\phi(\pi)\stackrel{\text{def}}{=}\mathsf{L}(\mathcal{A}_\pi\times\mathcal{B}_\pi).$$

Now, in order to prove the f-stage lemma, we apply Lemma 4.3 to the protocol Π with a different sub-additive measure. This measure takes into account both the complexity of rectangles of the form $\mathcal{A}_{\pi_1,Z}\times\mathcal{B}_{\pi_1,Z}$ and the number of Z's. We can therefore obtain many transcripts π_1 for which $\mathsf{L}(\mathcal{A}_{\pi_1,Z}\times\mathcal{B}_{\pi_1,Z})$ is large for many of the Z's, as required by the f-stage lemma.

We prove the f-stage lemma from Lemma 4.3 in Section 4.2, and then prove Lemma 4.3 in Section 4.3.

4.2 Proof of the f-stage lemma

Let us view Π as a tree, and its partial transcripts as vertices. We define the following measure on Π:

$$\phi(\pi)\stackrel{\text{def}}{=}\mathbb{E}_Z\left[\mathsf{L}(\mathcal{A}_{\pi,Z}\times\mathcal{B}_{\pi,Z})\right].$$

where the expectation is with respect to a uniformly chosen $Z\in\mathcal{Z}$. This measure is sub-additive since for every fixed Z, the measure $\mathsf{L}(\mathcal{A}_{\pi,Z}\times\mathcal{B}_{\pi,Z})$ is sub-additive. Furthermore, it holds that:

- ϕ assigns $\mathsf{L}(f)$ to the root of Π.
- For every leaf π of Π, it holds that $\phi(\pi)\leq 1$. The reason is that a leaf π must solve $KW_{f\diamond\oplus_n}$, and in particular must solve KW_f with respect to any Z that can reach it. Hence, $\mathsf{L}(\mathcal{A}_{\pi,Z}\times\mathcal{B}_{\pi,Z})\leq 1$.

We now apply Lemma 4.3 to Π and ϕ with $t = 2 \cdot 2^\ell$, and we get that there are at least

$$\frac{\mathsf{L}(f)}{(d+1) \cdot d \cdot 2 \cdot 2^\ell} = \frac{\mathsf{L}(f)}{O(d^2 \cdot 2^\ell)}$$

partial transcripts π_1 such that $\phi(\pi_1) \geq 2 \cdot 2^\ell$, none of them is an ancestor of another. We show that every such transcript π_1 is alive, and this will conclude the proof.

Let π_1 be partial transcript such that $\phi(\pi_1) \geq 2 \cdot 2^\ell$. In other words, it holds that

$$\mathbb{E}_{Z \in \mathcal{Z}} \left[\mathsf{L}(\mathcal{A}_{\pi,Z} \times \mathcal{B}_{\pi,Z}) \right] \geq 2 \cdot 2^\ell.$$

We now apply a standard averaging argument as follows. Since for any Z it holds that $\mathsf{L}(\mathcal{A}_{\pi,Z} \times \mathcal{B}_{\pi,Z}) \leq \mathsf{L}(f)$, there must be at least $2^\ell/\mathsf{L}(f)$ fraction of Z's for which $\mathsf{L}(\mathcal{A}_{\pi,Z} \times \mathcal{B}_{\pi,Z}) \geq 2^\ell$ (otherwise the expectation cannot reach $2 \cdot 2^\ell$). Since $2^\ell/\mathsf{L}(f) > 2^{-2m}$ we conclude that π_1 is $(\ell, 2^{-2m})$-alive, as required.

4.3 Proof of Lemma 4.3

Proof. Fix $t \in \mathbb{N}$. We can assume that $\phi(r) \geq (d+1) \cdot d \cdot t$ since otherwise there is nothing to prove. We say that a vertex v is a *maximal vertex* if $\phi(v) \leq d \cdot t$, and ϕ assigns to its parent a number that is greater than $d \cdot t$. We claim that T has at least $\phi(r)/d \cdot t$ maximal vertices: To see it, observe that there is a maximal vertex on every path from the root r to a leaf (since ϕ takes value t_0 at the leaves and at least $(d+1)dt > dt$ at the root). Hence, by the sub-additivity, if we denote by M the set of maximal vertices, we get that

$$\phi(r) \leq \sum_{v \in M} \phi(v) \leq d \cdot t \cdot |M| \,.$$

This implies that $|M| \geq \phi(r)/d \cdot t$, as required. We say that a maximal vertex v is *good* if $\phi(v) \geq t$, otherwise we say it is *bad*. We will prove that at least $1/(d+1)$ fraction of the maximal vertices are good, and this will imply the required result.

Let T' be the tree obtained by trimming T at maximal vertices – that is, for every maximal vertex v, we remove all the descendants of v and leave v as a leaf of T'. From now on, we refer to maximal vertices as *leaves* (since they are leaves of T'). In the new terminology, we wish to prove that at least $1/(d+1)$ fraction of the leaves of T' are good. We will prove it by constructing a d-to-1 mapping from the bad leaves to the good leaves. In order to construct this mapping, we use the following claim.

▶ **Claim 4.4.** *Every internal node of T' has at least one good leaf as a descendant.*

Proof. It suffices to prove the claim for internal nodes u such that $\phi(u) \leq 2 \cdot d \cdot t$, since every other internal node clearly has an internal descendant that satisfies this property.

Fix an internal node u such that $\phi(u) \leq 2 \cdot d \cdot t$. Now, observe that in the sub-tree rooted at u, every internal node has at least one child that is a leaf. In other words, this sub-tree looks like a path, with a leaf hanging from each vertex in the path. Therefore, this sub-tree contains at most d leaves. If all of those leaves were bad, then $\phi(u)$ would have been less than $d \cdot t$ (since by the sub-additivity, $\phi(u)$ is at most the sum of $\phi(v)$ for every leaf v in the sub-tree). However, we assumed that u is an internal node, and therefore $\phi(u)$ is greater than $d \cdot t$. Hence, at least one of the leaves must be good. ◀

Now, we define the mapping from the bad leaves to the good leaves as follows: Let v_{bad} be a bad leaf, and let u be the parent of v_{bad}. Then, we map v_{bad} to some arbitrarily chosen good leaf v_{good} that is a descendant of u – such a leaf v_{good} exists by the above claim.

We conclude the proof by showing that this mapping is d-to-1. Fix a good leaf v_{good}. Then, all the bad leaves that are mapped to v_{good} are direct children of the ancestors of v_{good}. Since T is of depth d, it follows that v_{good} has at most d ancestors, and therefore there are at most d bad leaves that are mapped to v_{good}. It follows that at least $1/(d+1)$ fraction of the leaves are good, as required. ◀

4.4 Generalized f-stage lemma

In this section, we prove a generalization of the f-stage lemma, that will be used in Section 8 below to prove the average-case version of the main theorem. While the f-stage lemma applies to protocols Π that solve $KW_{f\diamond\oplus_n}$, the generalization applies to protocols that only solve a sub-rectangle $\mathcal{X}\times\mathcal{Y}$ of $KW_{f\diamond\oplus_n}$, provided that $\mathcal{X}\times\mathcal{Y}$ has many "hard" f-rectangles. Recall that given a sub-rectangle $\mathcal{X}\times\mathcal{Y}$ of $KW_{f\diamond\oplus_n}$ and a product of edges Z, the f-rectangle of $\mathcal{X}\times\mathcal{Y}$ with respect to Z is the rectangle $\mathcal{A}\times\mathcal{B}$ defined by:

$$\begin{aligned}\mathcal{A} &\stackrel{\text{def}}{=} \left\{a\in f^{-1}(0) | Z^a\in\mathcal{X}\right\}\\ B &\stackrel{\text{def}}{=} \left\{b\in f^{-1}(1) | Z^b\in\mathcal{Y}\right\}.\end{aligned}$$

We have the following result.

▶ **Lemma 4.5** (generalized f-stage lemma). *Let $s\in\mathbb{N}$. Let $\mathcal{X}\times\mathcal{Y}$ be a sub-rectangle of $KW_{f\diamond\oplus_n}$ such that for at least 2^{-m} fraction of the Z's, the f-rectangle $\mathcal{A}\times\mathcal{B}$ of $\mathcal{X}\times\mathcal{Y}$ with respect to Z satisfies $\mathsf{L}(\mathcal{A}\times\mathcal{B})\geq s$. Let Π be a protocol for $KW_{\mathcal{X}\times\mathcal{Y}}$ of depth d. Then, there exist at least $s/O(2^\ell\cdot d^2)$ alive partial transcripts π_1 of Π, none of them is an ancestor of another.*

Proof. Let $\mathcal{Z}'$ be the set of Z's for which the f-rectangle $\mathcal{A}\times\mathcal{B}$ of $\mathcal{X}\times\mathcal{Y}$ with respect to Z satisfies $\mathsf{L}(\mathcal{A}\times\mathcal{B})\geq s$. As in the proof of the f-stage lemma in Section 4.2, we apply Lemma 4.3 to Π and ϕ where

$$\phi(\pi)\stackrel{\text{def}}{=}\mathbb{E}_{Z\in\mathcal{Z}'}\left[\mathsf{L}(\mathcal{A}_{\pi,Z}\times\mathcal{B}_{\pi,Z})\right].$$

We can lower bound the value that ϕ assigns to the root of Π by s, and upper bound each leaf by 1. We apply the lemma with $t=2\cdot 2^\ell$. We thus obtain that there are at least

$$\frac{s}{O(d^2\cdot 2^\ell)}$$

partial transcripts π_1 with $\phi(\pi_1)\geq 2\cdot 2^\ell$.

We now apply an averaging argument as before. Since for any $Z\in\mathcal{Z}'$ it holds that $\mathsf{L}(\mathcal{A}_{\pi,Z}\times\mathcal{B}_{\pi,Z})\leq\mathsf{L}(f)$, there must be at least $2^\ell/\mathsf{L}(f)$ fraction of Z's in $\mathcal{Z}'$ for which $\mathsf{L}(\mathcal{A}_{\pi,Z}\times\mathcal{B}_{\pi,Z})\geq 2^\ell$ (otherwise the expectation cannot reach 2ℓ). Since $2^\ell/\mathsf{L}(f)>2^{-m}$ we conclude that $\mathsf{L}(\mathcal{A}_{\pi,Z}\times\mathcal{B}_{\pi,Z})\geq 2^\ell$ for 2^{-m} fraction of $Z\in\mathcal{Z}'$ which is at least 2^{-2m} fraction of $\mathcal{Z}$. So π_1 is $(2^\ell,2^{-2m})$-alive as required. ◀

5 The Parity-Stage Lemma

In this section, we prove the parity-stage lemma, restated next. It is instructive to compare this proof to the proof of the lower bound for $KW_{\oplus_n}$ in Section 2.5.

▶ **Lemma 3.7** (restated). *Let Π_2 be a protocol that solves $KW_{f\diamond\oplus_n}$ on a t-almost hard distribution (X,Y) with probability 1. Then, Π_2 has at least $\mathsf{L}(\oplus_n)/2^{2t}$ transcripts.*

Proof. The basic idea of the proof is similar to that of the lower bound for $KW_{\oplus_n}$ in Section 2.5: At the end of the protocol, Alice and Bob must learn an axis j_i for some $i \in [m] - \mathcal{R}$, since the matrices X and Y differ only such axes. On the other hand, at the beginning of the protocol each of them knows at most t bits on each axis j_i, due to the definition of an almost hard distribution. Therefore, by the end of the protocol, each of the players has to learn at least $\log n - t$ bits of information, and the protocol must transmit at least $2\log n - 2t$ bits in total. The difference between this proof an the proof in Section 2.5 is that in the current proof, the players may choose which axis j_i they will learn among multiple options, and this complicates the argument a bit. Details follow.

Assume, for the sake of contradiction, that there is a protocol Π_2 that solves $KW_{f\diamond\oplus_n}$ on a t-almost hard distribution (X, Y) and is too efficient, i.e., has less than $\mathsf{L}(\oplus_n)/2^{2t}$ transcripts. Let $\pi_2 = \Pi_2(X, Y)$ be the (random) transcript of Π_2 when Alice and Bob get X and Y as inputs. By assumption, the support of π_2 is of size less than $\mathsf{L}(\oplus_n)/2^{2t} = n^2/2^{2t}$, and therefore

$$I(\pi_2 : X, Y) \leq H(\pi_2) < 2\log n - 2t.$$

On the other hand, by Fact 2.19 it holds that

$$I(\pi_2 : X, Y) \geq I(\pi_2 : X|Y) + I(\pi_2 : Y|X).$$

Hence, at least one of the terms on the right-hand side is smaller than $\log n - t$. Without loss of generality, assume that it is $I(\pi_2 : Y|X)$. It thus holds that

$$\log n - t > I(\pi_2 : Y|X) = H(\pi_2|X) - H(\pi_2|X, Y) = H(\pi_2|X),$$

where the last equality holds since X and Y determine π_2. Hence, there exists some specific X^* such that $H(\pi_2|X^*) < \log n - t$. Furthermore, since entropy is an upper-bound on min-entropy (Fact 2.18), it follows that $H_\infty(\pi_2|X^*) < \log n - t$. Therefore, there exists a specific transcript π_2^* such that $\log \frac{1}{\Pr[\pi_2^*|X^*]} < \log n - t$ or in other words,

$$\Pr[\pi_2^*|X^*] > \frac{2^t}{n}. \tag{2}$$

Suppose that this transcript π_2^* ends by outputting (i^*, j^*). Assuming the protocol solves $KW_{f\diamond\oplus_n}$, this means that for all X, Y's consistent with π_2^*, it holds that $X_{i^*,j^*} \neq Y_{i^*,j^*}$.

Now, let $\mathcal{R} \subseteq [m]$ be the set whose existence is guaranteed by the definition of an almost-hard distribution. We consider two cases, $i^* \in \mathcal{R}$ and $i^* \notin \mathcal{R}$, and show that in both cases there is a non-zero probability that $X_{i^*,j^*} = Y_{i^*,j^*}$ conditioned on π_2^*, thus obtaining a contradiction to the correctness of the protocol. Suppose first that $i^* \in \mathcal{R}$. In this case, it holds that $X_{i^*} = Y_{i^*}$ with probability 1. In particular, it follows that $X_{i^*,j^*} = Y_{i^*,j^*}$, as required.

Next, suppose that $i^* \notin \mathcal{R}$. This means that there is a random coordinate j_{i^*} such that either $X_{i^*} = Y_{i^*}$, or X_{i^*} and Y_{i^*} differ only on j_{i^*}. Moreover, it holds that

$$H_\infty(j_{i^*}|X^*) \geq \log n - t,$$

and in particular,

$$\Pr[j_{i^*} = j^*|X^*] \leq \frac{2^t}{n}. \tag{3}$$

By combining Inequalities 2 and 3, it holds that

$$\Pr\left[j_{i^*} = j^* | X^*, \pi_2^*\right] < 1.$$

The latter inequality implies that conditioned on X^* and π_2^*, the event "$j_{i^*} \neq j^*$" has non-zero probability. Now, observe that in this event it must hold that $X_{i^*,j^*} = Y_{i^*,j^*}$, since j_{i^*} is the only coordinate on which X_{i^*} and Y_{i^*} may differ. We conclude $X^*_{i^*j^*} = Y^*_{i^*j^*}$ with non-zero probability and this contradicts the correctness of the protocol. ◀

6 Technical tools

In this section we describe two technical tools that may be of independent interest.

The first is an averaging argument for min-entropy. Basically, it says that if we reveal $t \ll m$ bits of information on an m-tuple, then on most elements almost no information was revealed.

The second tool, which we call fortification, is a way to relate the information transmitted between Alice and Bob to the communication complexity of the residual problem. This is important because some of the steps we take in the proof of the main lemma reveal information to Alice and Bob, and we need to make sure that this does not decrease the complexity of the problem by too much.

6.1 An averaging argument for min-entropy

In our proof of the main lemma, we would like to say that if Alice and Bob communicated a small amount of information on the average row, then they communicated a small amount of information on *most* rows. This requires some sort of an averaging argument for information. Such an averaging argument is easy to prove for entropy, and was proved by [10] for an information measure called "predictability". In this section, we prove such an averaging argument for min-entropy.

On the high level, the averaging argument says that if at most r bits of information were communicated on a tuple $(u_1, \ldots, u_m)$ of random variables, then for every $k \geq 1$, at most $\frac{r}{k}$ bits of information were communicated on all but k of the random variables. As a warm-up, we first prove the following weak version of our averaging argument. We note that the following proof is similar to the proof of [10], and is also in the spirit of standard arguments from the literature on extractors.

▶ **Lemma 6.1** (Weak averaging argument for min-entropy). *Let $\mathcal{U}$ be some finite universe, and let $\overline{u} = (u_1, \ldots, u_m)$ be a tuple of random variables taking values in $\mathcal{U}$ such that $H_\infty(\overline{u}) \geq m \log|\mathcal{U}| - r$. Then, for every $k \geq 1$, there exists a set $\mathcal{R} \subseteq [m]$ of size at most k, and an event $E \subseteq \mathcal{U}^m$ of probability at least $|\mathcal{U}|^{-k}$, such that for every $i \in [m] - \mathcal{R}$ it holds that*

$$H_\infty(u_i | E) \geq \log|\mathcal{U}| - \frac{r}{k}.$$

Proof. In what follows, for every $\mathcal{S} \subseteq [m]$ we denote by $\overline{u}_\mathcal{S}$ the tuple of u_i's that belong to $\mathcal{S}$. We construct the set $\mathcal{R}$ and the event E iteratively. We start with $\mathcal{R} = \emptyset$ and $E = \mathcal{U}^m$. Then, in each iteration, if there is some $i \in [m] - \mathcal{R}$ that violates the above requirement, we add it to the set $\mathcal{R}$. More specifically, if i violates the requirement, then there is some specific value u_i^* such that

$$\Pr\left[u_i^* | E\right] \geq \frac{2^{r/k}}{|\mathcal{U}|}.$$

Then, we add i to $\mathcal{R}$, and add the condition $u_i = u_i^*$ to the event E (i.e., we set E to $E \cap \{\overline{u}' : u'_i = u_i^*\}$). The process stops when there is no $i \in [m] - \mathcal{R}$ that violates the requirement.

It remains to prove that $|\mathcal{R}| \leq k$. To this end, we prove that the following invariant is maintained throughout the iterations:

$$H_\infty(\overline{u}_{[m]-\mathcal{R}}|E) \geq (m - |\mathcal{R}|) \cdot \log|\mathcal{U}| - r + \frac{r}{k} \cdot |\mathcal{R}| \,.$$

This will imply the required upper bound on $|\mathcal{R}|$, since clearly the left-hand side cannot exceed $(m - |\mathcal{R}|) \cdot \log|\mathcal{U}|$.

We prove that the invariant is maintained by induction. First, note that it holds trivially when the process starts, i.e., when $\mathcal{R} = \emptyset$ and $E = \mathcal{U}^m$. Next, suppose that the invariant holds at the beginning of some iteration, and that in this iteration we add a coordinate i to $\mathcal{R}$. Then, for every assignment $\overline{u}^*_{[m]-(\mathcal{R}\cup\{i\})} \in \mathcal{U}^{[m]-(\mathcal{R}\cup\{i\})}$, it holds that

$$\begin{aligned}
& \Pr\left[\overline{u}_{[m]-(\mathcal{R}\cup\{i\})} = \overline{u}^*_{[m]-(\mathcal{R}\cup\{i\})} | E, u_i = u_i^*\right] \\
= \; & \frac{\Pr\left[\overline{u}_{[m]-(\mathcal{R}\cup\{i\})} = \overline{u}^*_{[m]-(\mathcal{R}\cup\{i\})} \text{ and } u_i = u_i^* | E\right]}{\Pr\left[u_i = u_i^* | E\right]} \\
\leq \; & 2^{-\left[(m-|\mathcal{R}|)\cdot\log|\mathcal{U}| - r + \frac{r}{k}\cdot|\mathcal{R}|\right]} / \Pr\left[u_i = u_i^* | E\right] \qquad (4) \\
\leq \; & 2^{-\left[(m-|\mathcal{R}|)\cdot\log|\mathcal{U}| - r + \frac{r}{k}\cdot|\mathcal{R}|\right]} / 2^{-(\log|\mathcal{U}| - r/k)} \qquad (5) \\
= \; & 2^{-\left[(m-|\mathcal{R}|-1)\cdot\log|\mathcal{U}| - r + \frac{r}{k}\cdot(|\mathcal{R}|+1)\right]},
\end{aligned}$$

where Inequality 4 holds due to the induction hypothesis, and Inequality 5 holds since i violates the requirement. This implies that

$$H_\infty(\overline{u}_{[m]-(\mathcal{R}\cup\{i\})}|E, u_i^*) \geq (m - |\mathcal{R}\cup\{i\}|) \cdot \log|\mathcal{U}| - r + \frac{r}{k} \cdot |\mathcal{R}\cup\{i\}| \,,$$

as required. Hence, it holds that $|\mathcal{R}| \leq k$ when the process ends. It is now easy to see that when the process ends, the probability of E is at least $\left(\frac{2^{r/k}}{|\mathcal{U}|}\right)^k \geq |\mathcal{U}|^{-k}$. The result follows. ◀

The reason we say that the above lemma is weak is because it only provides a lower bound of $|\mathcal{U}|^{-k}$ on the probability of the event E, which is very small when $\mathcal{U}$ is large. Intuitively, this means that in order to use this lemma, we need to reveal a lot of information to Alice and Bob. We therefore prove the following stronger version of the lemma that gives a lower bound of $m^{-O(k)}$, which is better when m is much smaller than $\mathcal{U}$, as is the case in our application. To the best of our knowledge, this stronger version of the lemma is new.

The basic idea of the proof is the following: whenever a coordinate i violates the requirement, it is because there was some "heavy" value u_i^*. In the above proof, we resolved this situation by conditioning on $u_i = u_i^*$, but this event may have a very low probability. In order to condition on an event with a higher probability, we consider two cases: If the heavy values, taken together, have relatively high probability, then we condition on the event that u_i takes a heavy value and add i to $\mathcal{R}$. If, on the other hand, the heavy values, taken together, have low probability, then we condition on u_i *not* taking a heavy value, and do *not* add i to $\mathcal{R}$ – hopefully, this will resolve the issue, because after discarding the heavy values, u_i will satisfy the requirement. This idea works, except for two minor issues:

- When we condition on i not taking a heavy value, this conditioning may cause new values to become heavy, even if they were not heavy before. This may get us into a "vicious cycle" of discarding values. In order to resolve this issue, whenever we discard heavy values, we increase the threshold that determines which values are considered heavy, so no new heavy values can be created immediately.
- When we condition u_i on any event – whether it is taking a heavy value or not taking a heavy value – it may cause new values to become heavy *for another random variable* $u_{i'}$. This may get us into a different "vicious cycle", in which we condition u_i, then condition $u_{i'}$, then condition u_i again, etc. In order to resolve this issue, we choose the different parameters such that u_i may cause new heavy values for another coordinate $u_{i'}$ only if u_i was conditioned on taking a heavy value. However, when u_i is conditioned on taking a heavy value, it is added to $\mathcal{R}$, and thus will not be selected again. Thus, the "vicious cycle" cannot happen.

We turn to provide the formal lemma and proof.

▶ **Lemma 6.2** (Averaging argument for min-entropy). *Let $\mathcal{U}$ be some finite universe, and let $\overline{u} = (u_1, \ldots, u_m)$ be a tuple of random variables taking values in $\mathcal{U}$ such that $H_\infty(\overline{u}) \geq m \log |\mathcal{U}| - r$. Then, for every $k \geq 1$, there exists a set $\mathcal{R} \subseteq [m]$ of size at most k, and an event E of probability at least $\frac{1}{4} \cdot m^{-2k}$, such that for every $i \in [m] - \mathcal{R}$ it holds that*

$$H_\infty(u_i|E) \geq \log |\mathcal{U}| - \frac{r+4}{k} - 2 \cdot \log m - 2.$$

Proof. For convenience, we denote

$$\tau \stackrel{\text{def}}{=} \log |\mathcal{U}| - \frac{r+4}{k} - 2 \cdot \log m,$$

that is, τ is the threshold of the lemma except for the additive term of -2.

We construct the set $\mathcal{R}$ and the event E iteratively. We start with $\mathcal{R} = \emptyset$ and $E = \mathcal{U}^m$. In each iteration, we select a coordinate $i \in [m]$ and do something with it. We describe a single iteration: Suppose that there is a coordinate $i \in [m] - \mathcal{R}$ that has been chosen in p previous iterations and that satisfies

$$H_\infty(u_i|E) \leq \tau + \log(1 - \frac{1}{m})^p.$$

Then, we select the coordinate i (the right-hand side is going to be the threshold that controls which values are considered "heavy"). By assumption, there exist values u_i^* such that

$$\Pr[u_i^*|E] \geq 2^{-\tau} / \left(1 - \frac{1}{m}\right)^p.$$

We define those values to be our "heavy values". Let E' be the event that u_i takes a heavy value. We consider two cases:

- If $\Pr[E'|E] \geq \frac{1}{m^2}$, then we set $E = E \cap E'$ and add i to $\mathcal{R}$.
- Otherwise, we set $E = E - E'$.

The following claim deals with the issues from the discussion above. Specifically, it shows that we chose the parameters in a way such that "new heavy values" can be created only by the first case above but not by the second case.

▶ **Claim 6.3.** *The second case above cannot occur twice for the same coordinate i without the first case occurring in between (for some index).*

Proof. Suppose otherwise. This means that there are some coordinate i and numbers $h_1 \leq h_2$ such that the second case occurred for i in both the h_1-th and h_2-th iterations, and the first case did not occur for any coordinate between those two iterations. Without loss of generality, we choose i, h_1, h_2 such that $h_2 - h_1$ is minimal among all the triplets (i, h_1, h_2) that satisfy those conditions.

Let p be the number of iterations in which i has been chosen before the h_1-th iteration. Let E_1 and E_2 be the event E at the h_1-th and the h_2-th iterations respectively. By assumption, only the second case happened for all the coordinates between those two iterations, and all those coordinates have been chosen at most once (because we assumed $h_2 - h_1$ is minimal). This implies that there have been at most m iterations between the h_1-th and h_2-th iterations, and in each of those iterations, the second case occurred. Now, observe that every time the second case occurs, the probability of E is multiplied by a factor that is at least $1 - \frac{1}{m^2}$. Therefore,

$$\Pr\left[E_2|E_1\right] \geq \left(1 - \frac{1}{m^2}\right)^m \geq 1 - \frac{1}{m}.$$

Let E_1' be the event E' at the h_1-th iteration, and observe that $E_2 \subseteq E_1 - E_1'$. Now, for every specific choice u_i^* of u_i in $E_1 - E_1'$, it holds that

$$\Pr\left[u_i^*|E_1\right] < 2^{-\tau} / \left(1 - \frac{1}{m}\right)^p.$$

Therefore, for every u_i^* it holds that

$$\begin{aligned}
\Pr\left[u_i^*|E_2\right] &= \frac{\Pr\left[u_i^* \wedge E_2|E_1\right]}{\Pr\left[E_2|E_1\right]} \\
&\leq \frac{\Pr\left[u_i^*|E_1\right]}{\Pr\left[E_2|E_1\right]} \\
&< \frac{2^{-\tau} / \left(1 - \frac{1}{m}\right)^p}{1 - \frac{1}{m}} \\
&= 2^{-\tau} / \left(1 - \frac{1}{m}\right)^{p+1}.
\end{aligned}$$

But this means that i could not have been selected at the h_2-th iteration, which is a contradiction. ◀

Observe that the first case cannot occur more than m times, and thus, combined with the latter claim, we get that the total number of iterations is at most m^2. In particular, the second case cannot happen for a coordinate i more than m times. Therefore, when the process terminates, every $i \in [m] - \mathcal{R}$ has been selected at most m times (since if $i \notin \mathcal{R}$, the first case never occurred for it). This implies that, when the process terminates, it holds for every $i \in [m] - \mathcal{R}$ that

$$\begin{aligned}
H_\infty(u_i|E) &\geq \tau + \log(1 - \frac{1}{m})^m \\
&\geq \tau - 2,
\end{aligned}$$

where the second inequality holds for sufficiently large m. This means that every $i \in [m] - \mathcal{R}$ satisfies the requirement of the lemma.

We turn to upper bounding the size of the set $\mathcal{R}$. Again, we do it by proving that an invariant is maintained throughout the iterations. Formally, we prove the following.

▶ **Claim 6.4.** *At each iteration, the following invariant is maintained:*

$$H_\infty(\overline{u}_{[m]-\mathcal{R}}|E) \geq (m-|\mathcal{R}|)\cdot\log|\mathcal{U}| - r + \frac{r+4}{k}\cdot|\mathcal{R}| - \frac{4\cdot s}{m^2},$$

where s is the total number of times the second case has occurred before this iteration.

Proof. We prove the claim by induction. Before the first iteration, when $|\mathcal{R}|=\emptyset$, the claim holds by the assumption of the lemma. Fix an iteration, let i be the coordinate that is selected in this iteration, and let s be the number of times the second case occurred before this iteration. We consider each of the two cases that may occur separately.

Suppose that the first case occurred, so $\Pr[E'|E] \geq \frac{1}{m^2}$. Then, for every assignment $\overline{u}^*_{[m]-(\mathcal{R}\cup\{i\})} \in \mathcal{U}^{[m]-(\mathcal{R}\cup\{i\})}$ the following holds:

$$\begin{aligned}
&\Pr\left[\overline{u}^*_{[m]-(\mathcal{R}\cup\{i\})}|E\cap E'\right]\\
=\;& \frac{\Pr\left[\overline{u}^*_{[m]-(\mathcal{R}\cup\{i\})} \text{ and } E'|E\right]}{\Pr[E'|E]}\\
\leq\;& m^2\cdot\Pr\left[\overline{u}^*_{[m]-(\mathcal{R}\cup\{i\})} \text{ and } E'|E\right]\\
=\;& m^2\cdot\sum_{u^*_i \text{ is a heavy value}}\Pr\left[\overline{u}^*_{[m]-(\mathcal{R}\cup\{i\})} \text{ and } u^*_i|E\right]\\
\leq\;& m^2\cdot\sum_{u^*_i \text{ is a heavy value}}2^{-H_\infty(\overline{u}_{[m]-\mathcal{R}}|E)}\\
\leq\;& m^2\cdot 2^{\tau}\cdot 2^{-H_\infty(\overline{u}_{[m]-\mathcal{R}}|E)} \qquad (6)\\
=\;& m^2\cdot\frac{2^{\log|\mathcal{U}|-(r+4)/k}}{m^2}\cdot 2^{-H_\infty(\overline{u}_{[m]-\mathcal{R}}|E)}\\
\leq\;& 2^{\log|\mathcal{U}|-(r+4)/k}\cdot 2^{-\left[(m-|\mathcal{R}|)\cdot\log|\mathcal{U}|-r+\frac{r+4}{k}\cdot|\mathcal{R}|-\frac{4\cdot s}{m^2}\right]} \qquad (7)\\
=\;& 2^{-\left[(m-|\mathcal{R}|-1)\cdot\log|\mathcal{U}|-r+\frac{r+4}{k}\cdot(|\mathcal{R}|+1)-\frac{4\cdot s}{m^2}\right]},
\end{aligned}$$

where Inequality 6 follows from the fact that there can be at most 2^τ heavy values, and Inequality 7 follows from the induction assumption. It follows that

$$H_\infty(\overline{u}_{[m]-(\mathcal{R}-\{i\})}) \geq (m-|\mathcal{R}\cup\{i\}|)\cdot\log|\mathcal{U}| - r + \frac{r+4}{k}\cdot|\mathcal{R}\cup\{i\}| - \frac{4\cdot s}{m^2},$$

as required.

Suppose now that the second case occurred. Then, for every assignment $\overline{u}^*_{[m]-\mathcal{R}} \in \mathcal{U}^{[m]-\mathcal{R}}$ it holds that

$$\begin{aligned}
\Pr\left[\overline{u}^*_{[m]-\mathcal{R}}|E-E'\right] &\leq \frac{\Pr\left[\overline{u}^*_{[m]-\mathcal{R}}|E\right]}{\Pr[E-E'|E]}\\
&\leq \frac{1}{1-\frac{1}{m^2}\cdot}\Pr\left[\overline{u}^*_{[m]-\mathcal{R}}|E\right]\\
&\leq \left(1+\frac{2}{m^2}\right)\cdot\Pr\left[\overline{u}^*_{[m]-\mathcal{R}}|E\right]\\
&\leq \exp(\frac{2}{m^2})\cdot\Pr\left[\overline{u}^*_{[m]-\mathcal{R}}|E\right]\\
&\leq 2^{-H_\infty(\overline{u}_{[m]-\mathcal{R}}|E)+\frac{4}{m^2}}\\
&\leq 2^{-\left[(m-|\mathcal{R}|)\cdot\log n-r+\frac{r+1}{k}\cdot|\mathcal{R}|-\frac{4\cdot(s+1)}{m^2}\right]},
\end{aligned}$$

where the last inequality follows from the induction assumption. It follows that

$$H_\infty(\overline{u}_{[m]-\mathcal{R}}) \geq (m - |\mathcal{R}|) \cdot \log n - r + \frac{r+4}{k} \cdot |\mathcal{R}| - \frac{4 \cdot (s+1)}{m^2},$$

as required. ◀

We can now bound the size of the set $\mathcal{R}$: since it must hold that $H_\infty(\overline{u}_{[m]-\mathcal{R}}|E) \leq (m - |\mathcal{R}|) \cdot \log n$, and since $s \leq m^2$, it follows from the last claim that $|\mathcal{R}| \leq k$, as required. It remains to lower bound the probability of the event E. The probability of E decreases by a factor of $\frac{1}{m^2}$ whenever the first case occurs, and by a factor of $1 - \frac{1}{m^2}$ whenever the second case occurs. The first case occurs at most k times, and the second case occurs at most m^2 times. Hence, the probability of E is at least

$$\left(\frac{1}{m^2}\right)^k \cdot (1 - \frac{1}{m^2})^{m^2} \geq \frac{1}{4} \cdot m^{2k},$$

as required. ◀

6.2 Fortification

In the proof of the main lemma, we will want to relate the information that Alice and Bob transmit about their inputs to the reduction in the complexity of the communication problem. For example, we will want to argue that if Alice and Bob transmitted only one bit of information, then the communication complexity of the problem was decreased by at most one bit (or, alternatively, that the protocol size of the problem was decreased by a factor of at most two).

However, this is not always true. For example, consider a KW relation $KW_{\mathcal{A}\times\mathcal{B}}$ (where $\mathcal{A}, B \subseteq \{0,1\}^m$ are disjoint), and suppose that the first bit of all the strings in $\mathcal{B}$ is 0, while in $\mathcal{A}$, the first bit is 0 for exactly half of the strings. In this case, if Alice tells Bob that the first bit of her input is 1, she only tells him only one bit of information, but the communication complexity of the problem drops to zero – since now Alice and Bob know that they differ on the first bit.

We say that a rectangle $\mathcal{A} \times \mathcal{B}$ is *fortified* [5] (with respect to a given protocol Π) if when Alice and Bob speak, the complexity is decreased in proportion to the information transmitted. More formally, we define fortified rectangles as follows.

▶ **Definition 6.5.** We say that a rectangle $\mathcal{A} \times \mathcal{B}$ is *ρ-fortified on Alice's side* if for every $\tilde{\mathcal{A}} \subseteq \mathcal{A}$ it holds that

$$\frac{\mathsf{L}(\tilde{\mathcal{A}} \times \mathcal{B})}{\mathsf{L}(\mathcal{A} \times \mathcal{B})} \geq \rho \cdot \frac{|\tilde{\mathcal{A}}|}{|\mathcal{A}|}.$$

Similarly, we say that $\mathcal{A} \times \mathcal{B}$ is *ρ-fortified on Bob's side* if the same holds for subsets $\tilde{\mathcal{B}} \subseteq \mathcal{B}$.

In this section, we show that even though there are rectangles $\mathcal{A} \times \mathcal{B}$ that are not fortified, every rectangle has a fortified sub-rectangle with similar complexity. For example, in the non-fortified rectangle $\mathcal{A} \times \mathcal{B}$ described above, we could take the sub-rectangle $\mathcal{A}' \times \mathcal{B}$ where $\mathcal{A}' \stackrel{\text{def}}{=} \{a \in \mathcal{A} : a_1 = 0\}$. More generally, we have the following result.

[5] *The term "fortified" was coined by Moshkovitz [26] in order to denote two-prover games that remain hard when restricted to sub-rectangles. She also proved a fortification lemma that transforms two-prover games into fortified ones. While our notion of fortification is very different from hers on the technical level, there is a conceptual similarity between the two notions.*

▶ **Lemma 6.6** (Fortification lemma). *Let $\mathcal{A}, \mathcal{B} \subseteq \{0,1\}^m$ be disjoint sets. There exists a subset $\mathcal{A}' \subseteq \mathcal{A}$ such that $\mathcal{A}' \times \mathcal{B}$ is $\frac{1}{4m}$-fortified on Alice's side, and such that $\mathsf{L}(\mathcal{A}' \times \mathcal{B}) \geq \frac{1}{4} \cdot \mathsf{L}(\mathcal{A} \times \mathcal{B})$. An analogous statement holds for Bob's side.*

▶ Remark. Although Definition 6.5 and Lemma 6.6 are phrased in terms of Karchmer-Wigderson relations, they work equally well for any communication problem.

We begin our proof of the fortification lemma by proving the following proposition, which is almost what we want.

▶ **Proposition 6.7.** *Let $\mathcal{A}, \mathcal{B} \subseteq \{0,1\}^m$ be disjoint sets. For every $0 < \rho < 1$, there exists $\mathcal{A}_1 \subseteq \mathcal{A}$ such that*

- *for every $\tilde{\mathcal{A}} \subseteq \mathcal{A}_1$ it holds that $\frac{\mathsf{L}(\tilde{\mathcal{A}} \times \mathcal{B})}{\mathsf{L}(\mathcal{A} \times \mathcal{B})} \geq \rho \cdot \frac{|\tilde{\mathcal{A}}|}{|\mathcal{A}|}$.*
- *$\mathsf{L}(\mathcal{A}_1 \times \mathcal{B}) \geq (1-\rho) \cdot \mathsf{L}(\mathcal{A} \times \mathcal{B})$.*

The same holds for $\mathcal{B}_1 \subseteq \mathcal{B}$.

The reason that Proposition 6.7 is not exactly what we want is that in the first item, the denominator on the right hand side is $|\mathcal{A}|$, while it should be $|\mathcal{A}_1|$ in order to satisfy the definition of a fortified rectangle. This is problematic, since in our application we will be able to control the ratio $\frac{|\tilde{\mathcal{A}}|}{|\mathcal{A}_1|}$, but we will have no way to control the ratio $\frac{|\tilde{\mathcal{A}}|}{|\mathcal{A}|}$.

Proof of Proposition 6.7. We prove the proposition for $\mathcal{A}_1 \subseteq \mathcal{A}$, and the proof for $\mathcal{B}_1 \subseteq \mathcal{B}$ is analogous. Let $\mathcal{A}_{\max} \subseteq \mathcal{A}$ be a maximal set that satisfies

$$\mathsf{L}(\mathcal{A}_{\max} \times \mathcal{B}) < \rho \cdot \frac{|\mathcal{A}_{\max}|}{|\mathcal{A}|} \cdot \mathsf{L}(\mathcal{A} \times \mathcal{B}). \tag{8}$$

We choose $\mathcal{A}_1 \stackrel{\text{def}}{=} \mathcal{A} - \mathcal{A}_{\max}$. Observe that it indeed holds that $\mathsf{L}(\mathcal{A}_1 \times \mathcal{B}) \geq (1-\rho) \cdot \mathsf{L}(\mathcal{A} \times \mathcal{B})$ by the sub-additivity of formula complexity. Now, suppose for the sake of contradiction that there is a set $\tilde{\mathcal{A}} \subseteq \mathcal{A}_1$ such that $\mathsf{L}(\tilde{\mathcal{A}}, \mathcal{B}) < \rho \cdot \frac{|\tilde{\mathcal{A}}|}{|\mathcal{A}|} \cdot \mathsf{L}(\mathcal{A}, \mathcal{B})$. Then, this would imply that

$$\begin{aligned}
\mathsf{L}\left((\tilde{\mathcal{A}} \cup \mathcal{A}_{\max}) \times \mathcal{B}\right) &\leq \mathsf{L}(\tilde{\mathcal{A}} \times \mathcal{B}) + \mathsf{L}(\mathcal{A}_{\max} \times \mathcal{B}) \\
&< \rho \cdot \frac{|\tilde{\mathcal{A}}|}{|\mathcal{A}|} \cdot \mathsf{L}(\mathcal{A} \times \mathcal{B}) + \rho \cdot \frac{|\mathcal{A}_{\max}|}{|\mathcal{A}|} \cdot \mathsf{L}(\mathcal{A} \times \mathcal{B}) \\
&= \rho \cdot \frac{|\tilde{\mathcal{A}} \cup \mathcal{A}_{\max}|}{|\mathcal{A}|} \cdot \mathsf{L}(\mathcal{A} \times \mathcal{B}),
\end{aligned}$$

where the first inequality holds by the sub-additivity of protocol size, and the second inequality holds by our assumptions on $\tilde{\mathcal{A}}$ and $\mathcal{A}_{\max}$. It follows that $\tilde{\mathcal{A}} \cup \mathcal{A}_{\max}$ is a set that satisfies Inequality 8 and that strictly contains $\mathcal{A}_{\max}$, thus contradicting the maximality of $\mathcal{A}_{\max}$. Hence, no such set $\tilde{\mathcal{A}}$ exists. ◀

▶ Remark. Consider again the example of a non-fortified rectangle $\mathcal{A} \times \mathcal{B}$ from the beginning of this section. For this rectangle, the above proof would take $\mathcal{A}_{\max}$ to be the set of strings a such that $a_1 = 1$. Thus, the set $\mathcal{A}'$ would be the set of strings a in which $a_1 = 0$, as required.

In order to prove the fortification lemma from Proposition 6.7, we need to replace the ratio $\frac{|\tilde{\mathcal{A}}|}{|\mathcal{A}|}$ with the ratio $\frac{|\tilde{\mathcal{A}}|}{|\mathcal{A}_1|}$. To this end, observe that

$$\frac{|\tilde{\mathcal{A}}|}{|\mathcal{A}_1|} = \frac{|\tilde{\mathcal{A}}|}{|\mathcal{A}|} / \frac{|\mathcal{A}_1|}{|\mathcal{A}|}.$$

Hence, we could achieve our goal by controlling the ratio $|\mathcal{A}_1|/|\mathcal{A}|$. The following proposition provides us the means to do so. Intuitively, this proposition is a form of "inverse fortification" – it allows us to lower bound the density of a subset $\tilde{\mathcal{A}}$ in terms of its complexity.

▶ **Proposition 6.8.** *Let $\mathcal{A}, \mathcal{B} \subseteq \{0,1\}^m$ be disjoint sets. For every $c \geq 1$, there exists a subset $\mathcal{A}_0 \subseteq \mathcal{A}$ such that for every $\tilde{\mathcal{A}} \subseteq \mathcal{A}_0$ it holds that*

$$\frac{|\tilde{\mathcal{A}}|}{|\mathcal{A}_0|} \geq \left(\frac{\mathsf{L}(\tilde{\mathcal{A}} \times \mathcal{B})}{\mathsf{L}(\mathcal{A}_0 \times \mathcal{B})}\right)^c, \tag{9}$$

and such that

$$\mathsf{L}(\mathcal{A}_0 \times \mathcal{B}) \geq 2^{-m/c} \cdot \mathsf{L}(\mathcal{A} \times \mathcal{B}). \tag{10}$$

Proof. We set $\mathcal{A}_0$ to be a minimal set that satisfies

$$\frac{|\mathcal{A}_0|}{|\mathcal{A}|} \leq \left(\frac{\mathsf{L}(\mathcal{A}_0 \times \mathcal{B})}{\mathsf{L}(\mathcal{A} \times \mathcal{B})}\right)^c.$$

Observe that $\mathcal{A}_0$ indeed satisfies Inequality 9: otherwise, there would have been $\tilde{\mathcal{A}} \subset \mathcal{A}_0$ that satisfied

$$\frac{|\tilde{\mathcal{A}}|}{|\mathcal{A}_0|} < \left(\frac{\mathsf{L}(\tilde{\mathcal{A}} \times \mathcal{B})}{\mathsf{L}(\mathcal{A}_0 \times \mathcal{B})}\right)^c,$$

and this would have implied that

$$\begin{aligned}
\frac{|\tilde{\mathcal{A}}|}{|\mathcal{A}|} &= \frac{|\tilde{\mathcal{A}}|}{|\mathcal{A}_0|} \cdot \frac{|\mathcal{A}_0|}{|\mathcal{A}|} \\
&< \left(\frac{\mathsf{L}(\tilde{\mathcal{A}} \times \mathcal{B})}{\mathsf{L}(\mathcal{A}_0 \times \mathcal{B})}\right)^c \cdot \left(\frac{\mathsf{L}(\mathcal{A}_0 \times \mathcal{B})}{\mathsf{L}(\mathcal{A} \times \mathcal{B})}\right)^c \\
&= \left(\frac{\mathsf{L}(\tilde{\mathcal{A}} \times \mathcal{B})}{\mathsf{L}(\mathcal{A} \times \mathcal{B})}\right)^c,
\end{aligned}$$

thus contradicting the minimality of $\mathcal{A}_0$. It remains to show that $\mathcal{A}_0 \times \mathcal{B}$ satisfies Inequality 10. It holds that

$$\frac{|\mathcal{A}_0|}{|\mathcal{A}|} \leq \left(\frac{\mathsf{L}(\mathcal{A}_0 \times \mathcal{B})}{\mathsf{L}(\mathcal{A} \times \mathcal{B})}\right)^c,$$

or in other words

$$\begin{aligned}
\mathsf{L}(\mathcal{A}_0 \times \mathcal{B}) &\geq \left(\frac{|\mathcal{A}_0|}{|\mathcal{A}|}\right)^{\frac{1}{c}} \cdot \mathsf{L}(\mathcal{A} \times \mathcal{B}) \\
&\geq \left(\frac{1}{2^m}\right)^{\frac{1}{c}} \cdot \mathsf{L}(\mathcal{A} \times \mathcal{B}) \\
&= 2^{-m/c} \cdot \mathsf{L}(\mathcal{A} \times \mathcal{B}),
\end{aligned}$$

as required. ◀

We are now ready to prove the fortification lemma.

Proof of Lemma 6.6. Let $\mathcal{A}, \mathcal{B} \subseteq \{0,1\}^m$ be disjoint sets. Our goal is to find a subset $\mathcal{A}' \subseteq \mathcal{A}$ such that $\mathcal{A}' \times \mathcal{B}$ is $\frac{1}{4m}$-fortified on Alice's side, and such that $\mathsf{L}(\mathcal{A}' \times \mathcal{B}) \geq \frac{1}{3} \cdot \mathsf{L}(\mathcal{A} \times \mathcal{B})$ (the proof for Bob's side is analogous). We start by applying Proposition 6.8 to $\mathcal{A} \times \mathcal{B}$ with $c = m$, thus obtaining a subset $\mathcal{A}_0 \subseteq \mathcal{A}$. Then, we apply Proposition 6.7 to $\mathcal{A}_0 \times \mathcal{B}$ with $\rho = \frac{1}{2m}$, thus obtaining a subset $\mathcal{A}_1 \subseteq \mathcal{A}_0$. Finally, we choose $\mathcal{A}'$ to be $\mathcal{A}_1$.

We prove that $\mathcal{A}'$ has the required properties. Observe that by Proposition 6.7, it holds that

$$\mathsf{L}(\mathcal{A}' \times \mathcal{B}) \geq (1 - \frac{1}{2m}) \cdot \mathsf{L}(\mathcal{A}_0 \times \mathcal{B}) \geq \frac{1}{2} \cdot \mathsf{L}(\mathcal{A}_0 \times \mathcal{B}),$$

and that by Proposition 6.8, it holds that

$$\mathsf{L}(\mathcal{A}_0 \times \mathcal{B}) \geq \frac{1}{2} \cdot \mathsf{L}(\mathcal{A} \times \mathcal{B}).$$

Therefore,

$$\mathsf{L}(\mathcal{A}' \times \mathcal{B}) \geq \frac{1}{4} \cdot \mathsf{L}(\mathcal{A} \times \mathcal{B}),$$

as required.

It remains to prove that $\mathcal{A}'$ is $\frac{1}{4m}$-fortified on Alice's side. Let $\tilde{\mathcal{A}} \subseteq \mathcal{A}'$. By Proposition 6.7, it holds that

$$\mathsf{L}(\tilde{\mathcal{A}} \times \mathcal{B}) \geq \frac{1}{2m} \cdot \frac{|\tilde{\mathcal{A}}|}{|\mathcal{A}_0|} \cdot \mathsf{L}(\mathcal{A}_0 \times \mathcal{B}) \geq \frac{1}{2m} \cdot \frac{|\mathcal{A}'|}{|\mathcal{A}_0|} \cdot \frac{|\tilde{\mathcal{A}}|}{|\mathcal{A}'|} \cdot \mathsf{L}(\mathcal{A}' \times \mathcal{B}).$$

Next, by Proposition 6.8, it holds that

$$\frac{|\mathcal{A}'|}{|\mathcal{A}_0|} \geq \left(\frac{\mathsf{L}(\mathcal{A}' \times \mathcal{B})}{\mathsf{L}(\mathcal{A}_0 \times \mathcal{B})}\right)^m \geq \left(1 - \frac{1}{2m}\right)^m \geq \frac{1}{2}.$$

Thus,

$$\mathsf{L}(\tilde{\mathcal{A}} \times \mathcal{B}) \geq \frac{1}{4m} \cdot \frac{|\tilde{\mathcal{A}}|}{|\mathcal{A}'|} \cdot \mathsf{L}(\mathcal{A}' \times \mathcal{B}).$$

The required result follows. ◀

7 Restating Lemma Proof of the Main Lemma

In this section, we prove the main lemma, restated next.

▶ **Lemma 3.6** (restated). *Let Π be a protocol for $KW_{f \diamond \oplus_n}$, and let π_1 be a live partial transcript of Π. Then, there exists a $\tilde{O}(\sqrt{m})$-almost hard distribution that is distributed over $\mathcal{X}_{\pi_1} \times \mathcal{Y}_{\pi_1}$.*

Fix a protocol Π for $KW_{f \diamond \oplus_n}$, and let π_1 be a live partial transcript of Π. Let $\mathcal{X}_{\pi_1} \times \mathcal{Y}_{\pi_1}$ be the rectangle associated with π_1. We would like to construct a t-almost hard distribution over $\mathcal{X}_{\pi_1} \times \mathcal{Y}_{\pi_1}$, where $t \stackrel{\text{def}}{=} \tilde{O}(\sqrt{m})$ and where the constant in the Big-O notation will be chosen to be sufficiently large as to make our argument hold.

7.1 Basic idea

By the definition of π_1 being alive, there is a set $\mathcal{Z}_{\pi_1} \subset \mathcal{Z}$, with $|\mathcal{Z}_{\pi_1}| \geq 2^{-2m} \cdot |\mathcal{Z}|$, such that for each $Z \in \mathcal{Z}_{\pi_1}$, it holds that $\mathsf{L}(\mathcal{A}_{\pi_1,Z} \times \mathcal{B}_{\pi_1,Z}) \geq 2^\ell$ where $\ell = C \cdot \sqrt{m} \cdot \log^C m$ for some large constant C to be determined later, and where

$$\begin{aligned} \mathcal{A}_{\pi_1,Z} &\stackrel{\text{def}}{=} \left\{a \in f^{-1}(0) | Z^a \in \mathcal{X}_{\pi_1}\right\} \\ B_{\pi_1,Z} &\stackrel{\text{def}}{=} \left\{b \in f^{-1}(1) | Z^b \in \mathcal{Y}_{\pi_1}\right\}. \end{aligned}$$

Consider the following graph G: the graph G is a layered graph, and the three layers are $\mathcal{X}_{\pi_1}$, $\mathcal{Z}_{\pi_1}$, and $\mathcal{Y}_{\pi_1}$. A vertex $X \in \mathcal{X}_{\pi_1}$ (respectively, $Y \in \mathcal{Y}_{\pi_1}$) is a neighbor of a vertex $Z = (Z^0, Z^1) \in \mathcal{Z}_{\pi_1}$ if and only if $X = Z^a$ for some $a \in f^{-1}(0)$ (respectively, $Y = Z^b$ for some $b \in f^{-1}(1)$). We define *the distribution* (X, Y) *of* G as the distribution that is sampled by picking a uniformly distributed path $X - Z - Y$ in G. While this distribution is not an almost-hard distribution, we will show that there is a subgraph G' of G such that the distribution *of* G' is an almost-hard distribution.

Let us examine the properties of the distribution (X, Y) of G more closely. Let Z denote the vertex that is sampled by this distribution, and let $j_1, \ldots, j_m \in [n]$ be the axes of Z (i.e., j_i is the unique coordinate on which Z_i^0 and Z_i^1 differ). Then, for every $i \in [m]$, it always holds that either $X_i = Y_i$, or that X_i and Y_i disagree exactly on one coordinate, which is j_i. Hence, in order for (X, Y) to be a t-almost hard distribution, it only needs to satisfy the property that for every $i \in [m]$, either that $X_i = Y_i$ with probability 1, or that for all specific choice $X^*$and Y^* of X and Y respectively, it holds that

$$\begin{aligned} H_\infty(j_i | X = X^*) &\geq \log n - t \\ H_\infty(j_i | Y = Y^*) &\geq \log n - t. \end{aligned}$$

This property would have been satisfied if $\mathcal{Z}_{\pi_1} = \mathcal{Z}$. In this case, $j_1, \ldots, j_m$ would have been uniformly distributed over $[n]$, and therefore all of them would have had min-entropy $\log n$ (conditioned on either X or Y). However, $\mathcal{Z}_{\pi_1}$ only constitutes 2^{-2m} fraction of $\mathcal{Z}$, and therefore the min-entropy of some j_i's may be as low as $\log n - 2m$. In order to resolve this issue, we apply the averaging argument for min-entropy (Lemma 6.2), and conclude the min-entropy of all but $\sqrt{m}$ of the j_i's is at least about $\log n - O(\sqrt{m})$. We refer to the $\sqrt{m}$ rows in which the min-entropy of j_i is lower than $\log n - O(\sqrt{m})$ as the *revealed rows*, and to the other rows as the *non-revealed rows*.

The non-revealed rows already satisfy what we need, so it remains to deal with the revealed rows. Let $\mathcal{R} \subseteq [m]$ denote the set of revealed rows. We will make sure that $X_i = Y_i$ for every $i \in \mathcal{R}$. To this end, recall that $X = Z^a$ and $Y = Z^b$ for some $a \in f^{-1}(0)$ and $b \in f^{-1}(1)$. We will construct the graph G' such that a and b always agree on the coordinates in $\mathcal{R}$.

To see why this is possible, recall that conditioned on the choice of Z, the strings a and b are taken from the rectangle $\mathcal{A}_{\pi_1,Z} \times \mathcal{B}_{\pi_1,Z}$. Since π_1 is alive with respect to Z, this means that $\mathsf{L}(\mathcal{A}_{\pi_1,Z} \times \mathcal{B}_{\pi_1,Z}) \geq 2^\ell$. We claim that this means that a and b can be chosen such that $a|_{\mathcal{R}} = b|_{\mathcal{R}}$. If this was not the case, i.e., if it was the case that $a|_{\mathcal{R}} \neq b|_{\mathcal{R}}$ for all $a \in \mathcal{A}_{\pi_1,Z}$ and $b \in \mathcal{B}_{\pi_1,Z}$, then the complexity of $\mathcal{A}_{\pi_1,Z} \times \mathcal{B}_{\pi_1,Z}$ would have been lower: Alice and Bob could have solved the game by sending each other their values at $\mathcal{R}$, and this protocol is of size at most $2^{O(|\mathcal{R}|)} < 2^\ell$ (for an appropriate choice of ℓ). There are two more complications:

- It is not sufficient to show that there exists at least one choice of a and b such that $a|_{\mathcal{R}} = b|_{\mathcal{R}}$. Rather, we need to show that there are many such choices – otherwise, forcing a and b to agree on $\mathcal{R}$ would reveal too much information to Alice and Bob.

  To this end, we process the graph as follows: we partition the strings $a \in \mathcal{A}_{\pi_1,Z}$ according

to $a|_{\mathcal{R}}$, and remove the classes that are too small. We do the same for $\mathcal{B}_{\pi_1,Z}$. By choosing the parameters appropriately, we can make sure that at most half of the strings in $\mathcal{A}_{\pi_1,Z}$ and $\mathcal{B}_{\pi_1,Z}$ are removed in the latter process. We then use the fortification lemma (Lemma 6.6) to show that the latter removal of strings did not decrease the complexity of $\mathcal{A}_{\pi_1,Z} \times \mathcal{B}_{\pi_1,Z}$ by too much, and hence this complexity is still large. We now argue as before that since $\mathcal{A}_{\pi_1,Z} \times \mathcal{B}_{\pi_1,Z}$ has large complexity, we can choose a class of $\mathcal{A}_{\pi_1,Z}$ and a class of $\mathcal{B}_{\pi_1,Z}$ that agree on $\mathcal{R}$.

Finally, we observe that the classes we chose must be large, since all the small classes were already removed. Hence, there are indeed many choices of a and b that agree on $\mathcal{R}$.

- The above discussion assumed implicitly that conditioned on X, the vertex Z is distributed uniformly over neighbors of X, and similarly for Y. This is may not always hold, but it does hold if all the vertices Z have the same degree. Throughout the proof, we take steps to ensure that the vertices Z have roughly equal degrees, and this will be good enough for our purposes.

7.2 A technical road-map

In the rest of this section, we describe the proof in detail. The proof follows the basic idea described above, but along the way there are some technical issues that need to be resolved and careful accounting that needs to be done. Therefore, we start by giving a "technical road-map" of the proof, which explains the main steps, the issues that we deal with, and the considerations that underly the accounting.

Notation

In what follows, we will consider subgraphs of the graph G described above. Given such a subgraph G_0 and a vertex Z, we define the *rectangle of Z in G_0* by $\mathcal{A}_{0,Z} \times B_{0,Z}$ where

$$\begin{aligned} \mathcal{A}_{0,Z} &\stackrel{\text{def}}{=} \left\{a \in f^{-1}(0) | Z^a \text{ is a neighbor of } Z \text{ in } G_0\right\} \\ B_{0,Z} &\stackrel{\text{def}}{=} \left\{b \in f^{-1}(1) | Z^b \text{ is a neighbor of } Z \text{ in } G_0\right\}. \end{aligned}$$

In general, we will identify the edges that come out of Z with the elements of $A_{0,Z}$ and $B_{0,Z}$. For example, we may say that we remove a string from $A_{0,Z}$ and mean that we remove the corresponding edge. We define the *complexity of Z in G_0* to be the protocol size of its rectangle, i.e., $\mathsf{L}(\mathcal{A}_{0,Z} \times B_{0,Z})$. Observe that in G, all the Z's have complexity at least 2^ℓ by the assumption that π_1 is alive.

Throughout the proof, we refer to the edges between $\mathcal{X}_{\pi_1}$ and $\mathcal{Z}$ as *the $\mathcal{X}$-side* of the graph or as *Alice's side* of the graph. Similarly, we refer to the edges between $\mathcal{Y}_{\pi_1}$ and $\mathcal{Z}$ as *the $\mathcal{Y}$-side* or as *Bob's side*.

7.2.1 The main steps

The proof consists of five main parts:

1. We process Alice's side, which means we remove vertices and edges in order to obtain some desired properties. Along the way, we construct the set $\mathcal{R}_{\mathcal{X}}$ of *revealed rows for the $\mathcal{X}$-side*, i.e., the set of indices $i \in [m]$ such that $H_\infty(j_i|X)$ is too small.
2. We process Bob's side in a similar manner, thus obtaining the set $\mathcal{R}_{\mathcal{Y}}$ of revealed rows for the $\mathcal{Y}$-side.
3. We force the X's and Y's in the graph to agree on the set of revealed rows $\mathcal{R} \stackrel{\text{def}}{=} \mathcal{R}_{\mathcal{X}} \cup \mathcal{R}_{\mathcal{Y}}$.

4. We perform a clean-up step that removes all the vertices whose degree became too small, and denote the resulting graph by G'.
5. We conclude by proving that the distribution of G' is an almost-hard distribution, as required.

The most technical part is the processing of Alice's side. It consists of four steps:

- **Fortification:** We fortify the the rectangle of each Z. This is done to make sure that the following steps, which remove edges on Alice's side, do not reduce the complexity of the Z's by too much. This results in a subgraph of G that we denote by G_{A1} (here, "$A1$" denotes "first step on Alice's side").
- **Regularization:** As discussed above, throughout the proof we will need to guarantee that the Z's are *roughly regular*, i.e., that all the Z's have roughly the same degree. We create this property in this step, by taking a subset of the Z's that have roughly the same degree on the $\mathcal{X}$-side in G_{A1}, and discarding all the rest. We denote the resulting subgraph by G_{A2}.
- **Finding the revealed rows:** For each X in G_{A2}, we consider the distribution on axes $j_1, \ldots, j_m$ that is induced by choosing a random neighbor Z of X. We observe that this distribution has min-entropy which is at least $m \cdot \log n - O(m)$, and apply the averaging argument for min-entropy to this distribution (Lemma 6.2). This yields a set of revealed axes $\mathcal{R}_X$ of size $\sqrt{m}$, such that the min-entropy of each j_i for $i \in [m] - \mathcal{R}_X$ is at least $\log n - \tilde{O}(\sqrt{m})$.

  Note that the averaging argument only says that the min-entropy of j_i is large conditioned on some event E_X. We therefore remove from the graph all the edges that are not consistent with E_X, for each X. In addition, note that the set $\mathcal{R}_X$ may be different for each X. We now choose the most popular set $\mathcal{R}_X$, denote it by $\mathcal{R}_{\mathcal{X}}$, and discard all the X's with a different set. We denote the resulting subgraph by G_{A3}.
- **Removing the small classes:** For each Z, consider its rectangle $\mathcal{A}_{A3,Z} \times \mathcal{B}_{A3,Z}$ in G_{A3}. We would like to partition the strings $a \in \mathcal{A}_{A3}$ into classes according to $a|_{\mathcal{R}}$, and remove the small classes – as discussed above, this is done in order to make sure that when we force the a's and the b's to agree on $\mathcal{R}$, we will retain many a's.

  However, there is a small issue here that needs to be dealt with: at this point we do not know yet the set $\mathcal{R}$ of revealed rows – we know the set $\mathcal{R}_{\mathcal{X}}$ of revealed rows for the $\mathcal{X}$-side, but we do not know yet the set $\mathcal{R}_{\mathcal{Y}}$ of revealed rows for the $\mathcal{Y}$-side. Therefore, we perform this step of "removing the small classes" for every possible candidate for $\mathcal{R}$. By choosing the parameters appropriately, we can ensure that doing so does not remove too many a's. We denote the resulting subgraph by G_{A4}.

The processing of Bob's side is similar to that of Alice's side, except that the step of removing the small classes is a little simpler since at this point we know $\mathcal{R}$. This processing creates corresponding subgraphs $G_{B1}, G_{B2}, G_{B3}, G_{B4}$.

Next, we force the X's and Y's to agree on the revealed rows as follows: For every Z in G_{B4}, we consider the rectangle $\mathcal{A}_{B4,Z} \times \mathcal{B}_{B4,Z}$. We claim that there must be $a \in \mathcal{A}_{B4,Z}$ and $b \in \mathcal{B}_{B4,Z}$ such that $a|_{\mathcal{R}} = b|_{\mathcal{R}}$, or otherwise the complexity of $\mathcal{A}_{B4,Z} \times \mathcal{B}_{B4,Z}$ would have been too small. We then claim that a and b must belong to large classes of $\mathcal{A}_{B4,Z}$ and $\mathcal{B}_{B4,Z}$ respectively, since the small classes have already been removed, and therefore there are *many* a's and b's such that $a|_{\mathcal{R}} = b|_{\mathcal{R}}$. We now discard all the other a's and b's for every Z, thus creating a new subgraph G_{agr}.

The final step is the clean-up step. The reason that this step is needed is that each of the previous steps removed some edges. This is problematic for two reasons: First, the degree of some X's may have become too small, in which case the min-entropy $H_\infty(j_i|X)$ may also

become too small, and the same goes for the Y's. Second, the degree of some Z's may have become too small, thus violating the rough regularity of the Z's. In order to rectify those violations, we remove the vertices whose degrees are too small. However, this removal may decrease the degrees of other vertices, so we continue removing vertices until there are no more vertices whose degrees are too small. By choosing the parameters appropriately, we can make sure that the process terminates before the whole graph is deleted.

7.2.2 Issues and accounting

Retaining a large number of edges

Recall that at the end of the step of "finding the revealed rows" on Alice's side, we have for each X the property that for every $i \in [m] - \mathcal{R}_{\mathcal{X}}$, it holds that

$$H_\infty(j_i|X) \geq \log n - \tilde{O}(\sqrt{m}).$$

However, in the following steps, we remove vertices and edges from the graph, and this may destroy this property. More specifically, after we remove edges from the graph, this property will continue to hold for every X whose degree was reduced by a factor of at most $2^{\tilde{O}(\sqrt{m})}$, but may cease to hold for X's whose degree was reduced by more than that.

As explained above, we deal with this issue in the clean-up step by removing all the X's whose degree is too small, i.e., whose degree was reduced by a factor of more than $2^{\tilde{O}(\sqrt{m})}$. However, in order for this solution to be effective, we need to make sure that *the degree of most X's is not too small* (or otherwise the clean-up may remove too many X's).

To this end, it suffices to show that the number of edges of G_{agr} on the $\mathcal{X}$-side is at least $2^{-\tilde{O}(\sqrt{m})}$ times the number of edges of G_{A3} on the $\mathcal{X}$-side. In order to do so, we keep track of the number of edges on the $\mathcal{X}$-side throughout the proof and make sure that it does not decrease too much. The same goes for the $\mathcal{Y}$-side.

Retaining a large number of Z's

When we perform the step of "finding the revealed rows" on Alice's side, we use the fact that in G_{A2}, the distribution $j_1, \ldots, j_m$ has min-entropy at least

$$m \cdot \log(n) - O(m).$$

In order to show this lower bound on the min-entropy, we use the fact that the number of Z's in G_{A2} is at least $2^{-O(m)}$ fraction of all the possible Z's. The latter fact follows from the assumption that π_1 is alive, but we also need to make sure that it is not invalidated by the regularization step. Therefore, when performing the regularization, we make sure that we did not remove too many Z's.

Furthermore, since we also perform the step of "finding the revealed rows" on Bob's side, we also need to make sure that the number of Z's in the graph G_{B2} is sufficiently large. To this end, we keep track of the number of Z's throughout the processing on Alice's side and make sure that we do not remove too many Z's.

Interaction between the two sides of the graph

When we process Bob's side, we remove some of the Z's in the regularization step and in the step of "removing the small classes". However, when we remove Z's, it also causes the removal of edges on the $\mathcal{X}$-side. Hence, we have to make sure that those steps do not remove too many edges on the $\mathcal{X}$-side.

To this end, we first make sure that those steps do not remove too many Z's: in particular, we make sure that after each step, we retain at least $2^{-\tilde{O}(\sqrt{m})}$ fraction of the Z's. Then, we use the fact that the Z's are roughly regular on the $\mathcal{X}$-side to deduce that we retained at least $2^{-\tilde{O}(\sqrt{m})}$ fraction of the edges on the $\mathcal{X}$-side.

Average degree vs. minimum degree

In many places throughout the proof, we will have a lower bound on the *average degree* of vertices, but we will want this lower bound to hold for the *minimum degree*, i.e., we will want it to hold for every vertex. For example, at the beginning of the step of "finding the revealed rows" on Alice's side, we know that *the average* X is connected to at least $2^{-O(m)}$ fraction of all the possible Z's, but we will want it to hold *for every* X. Whenever we encounter such a situation, we resolve the issue by removing from the graph all the vertices whose degree is too small compared to the average degree. We will use the following fact to show that this removal does not discard too many edges.

▶ **Fact 7.1.** *Let $G_0 = (\mathcal{U}_0 \cup \mathcal{V}_0, \mathcal{E}_0)$ be a bipartite graph, and denote the average degree of $\mathcal{U}_0$ by $d_{\mathcal{U}}$. If we remove all the vertices of $\mathcal{U}_0$ whose degree is less than $\varepsilon \cdot d_{\mathcal{U}}$, then we remove at most ε fraction of the total number of edges.*

Proof. By the definition of average degree, it holds that $|\mathcal{E}_0| = d_{\mathcal{U}} \cdot |\mathcal{U}_0|$. The number of vertices that we remove is at most $|\mathcal{U}_0|$, and each of them is connected to at most $\varepsilon \cdot d_{\mathcal{U}}$ edges. Hence, the total number of edges we removed is at most $\varepsilon \cdot d_{\mathcal{U}} \cdot |\mathcal{U}_0| = \varepsilon \cdot |\mathcal{E}_0|$, as required. ◀

We finally turn to present the full proof.

7.3 Processing Alice's side

Fortification

The first step we take in processing the graph on Alice's side is fortifying the Z's on Alice's side. For each Z, we apply the fortification lemma (Lemma 6.6) to the rectangle of Z in G, namely $\mathcal{A}_{\pi_1,Z} \times \mathcal{B}_{\pi_1,Z}$, thus obtaining a sub-rectangle $\mathcal{A}_{A1,Z} \times \mathcal{B}_{A1,Z}$ that is $\frac{1}{4m}$-fortified on Alice's side (where $\mathcal{B}_{A1,Z} = \mathcal{B}_{\pi_1,Z}$). We then replace $A_{\pi_1,Z} \times B_{\pi_1,Z}$ with $\mathcal{A}_{A1,Z} \times \mathcal{B}_{A1,Z}$ by removing from G all the edges that correspond to strings in $\mathcal{A}_{\pi_1,Z} - \mathcal{A}_{A1,Z}$. We denote the resulting graph by G_{A1}.

Regularization

Next, we make sure that all the vertices Z have roughly the same degree on the $\mathcal{X}$-side (i.e., have the same number of neighbors X). To this end, we partition the Z's to $m+1$ classes, such that the Z's in the i-th class has degree at least 2^{i-1} and less than 2^i (for $1 \leq i \leq m+1$). Let i be such that the i-th class is the class that contains a largest number of Z's. We remove from G_{A1} all the Z's outside the i-th class, and denote the resulting graph by G_{A2} and the resulting set of Z's by $\mathcal{Z}_{A2}$.

Let $d_{\mathcal{Z},\mathcal{X}} \stackrel{\text{def}}{=} 2^i$. By definition, all the vertices Z in $\mathcal{Z}_{A2}$ have degrees between $\frac{1}{2} \cdot d_{\mathcal{Z},\mathcal{X}}$ and $d_{\mathcal{Z},\mathcal{X}}$. Moreover, observe that G_{A2} retains at least $\frac{1}{m+1}$ fraction of the Z's. Since G originally had at least $2^{-2m} \cdot |\mathcal{Z}|$ vertices Z (and so did G_{A1}), it follows that G_{A2} has at least $2^{-2m-\log(m+1)} \cdot |\mathcal{Z}|$ vertices Z.

Finding the revealed rows

We turn to applying the averaging argument to the X's in order to find the revealed rows. However, we can only do so for X's with sufficiently large degree. To compute the average degree of the X's we observe that each Z must be connected to at least one vertex X, and therefore the average degree of the X's is at least

$$\frac{|\mathcal{Z}_{A2}|}{|\mathcal{X}_{\pi_1}|} \geq \frac{2^{-2m-\log(m+1)} \cdot |\mathcal{Z}|}{2^{m \cdot n}} = \frac{2^{-2m-\log(m+1)} \cdot \left(2^{m \cdot (n-1)} \cdot n^m\right)}{2^{m \cdot n}} = 2^{-3m-\log(m+1)} \cdot n^m.$$

We remove from the graph all the X's with degree less than $2^{-4m} \cdot n^m$. By Fact 7.1, we removed less than half of the edges of the graph on the $\mathcal{X}$-side.

Now, for each of the remaining $X's$, we perform the following steps. Let Z be a uniformly distributed neighbor of X, and let $j_1, \ldots, j_m$ be the axes of Z. Observe that given X, there is a one-to-one correspondence between Z and the sequence $j_1, \ldots, j_m$. Thus, the fact that the degree of X is at least $2^{-4m} \cdot n^m$ implies that

$$H_\infty(j_1, \ldots, j_m) \geq m \cdot \log n - 4 \cdot m.$$

We apply the averaging argument for min-entropy (Lemma 6.2) to $j_1, \ldots, j_m$ with parameters $r = 4m$ and $k = \sqrt{m}$, thus obtaining a set $\mathcal{R}_X$ of size $\sqrt{m}$ and an event $E_X \subseteq [n]^m$ of probability at least $2^{-O(\sqrt{m} \log m)} = 2^{-\tilde{O}(\sqrt{m})}$ such that for every $i \in [m] - \mathcal{R}_X$ it holds that

$$H_\infty(j_i | E_X) \geq \log n - O(\sqrt{m}).$$

Observe that the event E_X is a set of tuples $(j_1, \ldots, j_m)$, each of which corresponds to an edge going out of X. We remove all the edges of X that do not belong to E_X. Note that we retain at least $2^{-\tilde{O}(\sqrt{m})}$ fraction of the edges since the probability of E_X is at least $2^{-\tilde{O}(\sqrt{m})}$.

Next, we partition the X's according to their set $\mathcal{R}_X$, pick the class that is connected to the largest number of edges, and remove all the X's outside of this class. Let $\mathcal{R}_\mathcal{X}$ be the set R_X of the class that was picked, and denote by G_{A3} the resulting graph. There are $\binom{m}{\sqrt{m}} = 2^{\tilde{O}(\sqrt{m})}$ classes so it is easy to see that after the removal we retain at least $2^{-\tilde{O}(\sqrt{m})}$ fraction of the edges, and therefore G_{A3} retains at least $2^{-\tilde{O}(\sqrt{m})}$ fraction of the edges of G_{A2}.

Summing up the discussion so far, the graph G_{A3} has the following property: Let X be a vertex in G_{A3}, let Z be a uniformly distributed neighbor of X in G_{A3}, and let $j_1, \ldots, j_m$ be the axes of the edges in Z. Then, for every $i \in [m] - \mathcal{R}_\mathcal{X}$ it holds that

$$H_\infty(j_i) \geq \log n - O(\sqrt{m}). \tag{11}$$

Removing small classes from the rectangles of the Z's

The last step we perform is a preparation toward forcing the a's and the b's of each Z to agree on the revealed rows – see the discussion in Section 7.1 about the first complication. As explained there, for each Z, we would like to partition its set of a's according to their values at the revealed rows $\mathcal{R}$, and remove the classes of the partition that are too small.

However, we do not know yet what is the set $\mathcal{R}$ of revealed rows. Indeed, we know the set $\mathcal{R}_\mathcal{X}$ of the revealed rows on Alice's side, but we do not know yet the revealed rows on Bob's side. In order to resolve this issue, we define classes of edges *for all the possible candidates for* $\mathcal{R}$, and remove the small classes. Note that now the classes no longer form a partition of the a's of Z, but it does not matter for our argument.

Formally, we define a *label* to be a pair $(\mathcal{R}, \lambda)$ where $\mathcal{R} \subseteq [m]$ is a set of size $2\sqrt{m}$ that contains $\mathcal{R}_{\mathcal{X}}$, and $\lambda \in \{0,1\}^{\mathcal{R}}$ is an assignment of bits to $\mathcal{R}$. There are only $2^{\tilde{O}(\sqrt{m})}$ possible labels. We say that a string $a \in \{0,1\}^m$ is *consistent with the label* $(\mathcal{R}, \lambda)$ if $a|_{\mathcal{R}} = \lambda$.

Next, we perform the following for each vertex Z in G: Let $\mathcal{A}_{A3,Z} \times \mathcal{B}_{A3,Z}$ be the rectangle of Z in G_3. For every possible label $(\mathcal{R}, r)$, define *the class of* $(\mathcal{R}, \lambda)$ to be the subset of all strings $a \in \mathcal{A}_{A3,Z}$ that are consistent with $(\mathcal{R}, \lambda)$. We say that a class is *small* if it contains less than $2^{-3 \cdot \sqrt{m} \cdot \log m}$ fraction of the strings in $\mathcal{A}_{A3,Z}$. We now remove from $\mathcal{A}_{A3,Z}$ every string a that belongs to some small class. If new small classes are created by the latter removal, we remove them as well, and repeat this process until no small classes remain. By the union bound, it is not hard to see that this removes at most half of the strings in $\mathcal{A}_{A3,Z}$. Denote the resulting set by $\mathcal{A}_{A4,Z}$, and let $\mathcal{B}_{A4,Z} \stackrel{\text{def}}{=} \mathcal{B}_{A3,Z}$.

Finally, observe that the average degree of the Z's on the $\mathcal{X}$-side is at least $2^{-\tilde{O}(\sqrt{m})} \cdot d_{\mathcal{Z},\mathcal{X}}$: After the regularization, the average degree was at least $\frac{1}{2} \cdot d_{\mathcal{Z},\mathcal{X}}$, and after finding the revealed rows and removing the small classes we retained at least $2^{-\tilde{O}(\sqrt{m})}$ fraction of the edges. We now remove all the $Z's$ whose degree is less than half the average degree in order to maintain the property that all the Z's have roughly the same degree – in particular, after the removal, all Z's will have degree between $2^{-\tilde{O}(\sqrt{m})} \cdot d_{\mathcal{Z},\mathcal{X}}$ and $d_{\mathcal{Z},\mathcal{X}}$. We denote the resulting set of Z's by $\mathcal{Z}_{A4}$, and the resulting graph by G_{A4}.

Observe that G_{A4} retains quarter of the edges of G_{A3} on the $\mathcal{X}$-side: The removal of the small classes removed at most half of the edges of each Z, and hence at most half of the edges of G_{A3}. Then, the removal of low-degree Z's removed at most half of the remaining edges by Fact 7.1. Since G_{A3} retained $2^{-\tilde{O}(\sqrt{m})}$ fraction of the edges of G_{A2}, it follows that G_{A4} retains $2^{-\tilde{O}(\sqrt{m})}$ fraction of the edges of G_{A2}.

Furthermore, we claim that the set $\mathcal{Z}_{A4}$ of Z's in G_{A4} contains at least $2^{-\tilde{O}(\sqrt{m})}$ fraction of the Z's in $\mathcal{Z}_{A2}$. To see why this is the case, recall that the number of edges on the $\mathcal{X}$-side in G_{A2} is at least $\frac{1}{2} \cdot d_{\mathcal{Z},\mathcal{X}} \cdot |\mathcal{Z}_{A,2}|$ (since the minimal degree of a Z in G_{A2} is $\frac{1}{2} \cdot d_{\mathcal{Z},\mathcal{X}}$). On the other hand, the number of edges on the $\mathcal{X}$-side in G_{A4} is at most $d_{\mathcal{Z},\mathcal{X}} \cdot |\mathcal{Z}_{A4}|$, and we know that this number is at least $2^{-\tilde{O}(\sqrt{m})}$ fraction of the number of edges in G_{A2}. Therefore,

$$\begin{aligned}
d_{\mathcal{Z},\mathcal{X}} \cdot |\mathcal{Z}_{A4}| &\geq 2^{-\tilde{O}(\sqrt{m})} \cdot \frac{1}{2} \cdot d_{\mathcal{Z},\mathcal{X}} \cdot |\mathcal{Z}_{A,2}| \\
|\mathcal{Z}_{A4}| &\geq 2^{-\tilde{O}(\sqrt{m})} \cdot |\mathcal{Z}_{A,2}| \\
&\geq 2^{-\tilde{O}(\sqrt{m})} \cdot |\mathcal{Z}_{A,1}| \\
&\geq 2^{-O(m)} \cdot |\mathcal{Z}| \,.
\end{aligned}$$

Moreover, observe that the complexity of every $Z \in \mathcal{Z}_{A4}$ is at least $2^{-\tilde{O}(\sqrt{m})}$ fraction of its original complexity in G: First, recall that the complexity of the fortified rectangles $\mathcal{A}_{A1,Z} \times \mathcal{B}_{A1,Z}$ was $\frac{1}{3}$ fraction of the original complexity. By the fortification, the complexity of each Z in G_4 is

$$\begin{aligned}
\mathsf{L}(\mathcal{A}_{A4,Z} \times \mathcal{B}_{A4,Z}) &\geq \frac{1}{4m} \cdot \frac{|\mathcal{A}_{A4,Z}|}{|\mathcal{A}_{A1,Z}|} \cdot \mathsf{L}(\mathcal{A}_{A1,Z} \times \mathcal{B}_{A1,Z}) \\
&\geq 2^{-\tilde{O}(\sqrt{m})} \cdot \mathsf{L}(\mathcal{A}_{A1,Z} \times \mathcal{B}_{A1,Z}) \\
&\geq 2^{\ell - \tilde{O}(\sqrt{m})}.
\end{aligned}$$

7.4 Processing Bob's side

We now take the same steps as in Section 7.3 in the $\mathcal{Y}$-side of the graph: We apply the fortification on Bob's side to the vertices Z in G_{A4}, thus obtaining a new graph G_{B1}. We

then apply regularization, thus obtaining a new graph G_{B2} such that the degrees of the Z's on the $\mathcal{Y}$-side are are between $\frac{1}{2} \cdot d_{\mathcal{Z},\mathcal{Y}}$ and $d_{\mathcal{Z},\mathcal{Y}}$ for some degree $d_{\mathcal{Z},\mathcal{Y}}$. Next, we find the revealed rows for the Y's, thus obtaining a new graph G_{B3} and a set $\mathcal{R}_{\mathcal{Y}}$ such that the following holds for every Y: Let Z be a uniformly distributed neighbor of Y, and let $j_1, \ldots, j_m$ be the axes of Z. Then, for every $i \in [m] - \mathcal{R}_{\mathcal{Y}}$ it holds that

$$H_\infty(j_i) \geq \log n - \tilde{O}(\sqrt{m}). \tag{12}$$

Let $\mathcal{R} \stackrel{\text{def}}{=} \mathcal{R}_{\mathcal{X}} \cup \mathcal{R}_{\mathcal{Y}}$.

There is a small difference in the step of "removing the small classes": Now, we know the set of revealed rows $\mathcal{R}$, so we do not need the labels to contain a candidate for $\mathcal{R}$. Instead, for each Z, we simply partition the strings $b \in \mathcal{B}_{B3,Z}$ according to $b|_{\mathcal{R}}$, and remove all the classes that contain only $2^{-3\cdot\sqrt{m}}$ fraction of the strings in $\mathcal{B}_{B3,Z}$. The rest of this step proceeds as before, and we denote the resulting graph by G_{B4}.

Again, we note that the following points:

- The graph G_{B4} retains at least $2^{-\tilde{O}(\sqrt{m})}$ fraction of the vertices Z of G_{A4}.
- The degree of every Z in G_{B4} on the $\mathcal{Y}$-side is at least $2^{-\tilde{O}(\sqrt{m})} \cdot d_{\mathcal{Z},\mathcal{Y}}$.
- The complexity of each Z in G_{B4} is at least $2^{\ell - \tilde{O}(\sqrt{m})}$.
- The graph G_{B4} retains at least $2^{-\tilde{O}(\sqrt{m})}$ fraction of the edges of G_{B3} on the $\mathcal{Y}$-side.

It is also important to note that G_{B4} does not lose too many edges on the $\mathcal{X}$-side: We lose edges on the $\mathcal{X}$-side when we remove Z's. However, since $|\mathcal{Z}_{B4}| \geq 2^{-\tilde{O}(\sqrt{m})} \cdot |\mathcal{Z}_{A4}|$, and since all the degrees of Z's on the $\mathcal{X}$-side are between $2^{-\tilde{O}(\sqrt{m})} \cdot d_{\mathcal{Z},\mathcal{X}}$ and $d_{\mathcal{Z},\mathcal{X}}$, the graph G_{B4} retains at least $2^{-\tilde{O}(\sqrt{m})}$ fraction of the edges of G_{A4} on the $\mathcal{X}$-side.

7.5 Forcing agreement on the revealed rows

We are now ready to force the a's and b's of each Z to agree on $\mathcal{R}$. Fix a vertex Z. We show that there exists an assignment $\lambda_z \in \{0,1\}^{\mathcal{R}}$, and strings $a \in \mathcal{A}_{B4,Z}$ and $b \in \mathcal{B}_{B4,Z}$ such that

$$a|_{\mathcal{R}} = b|_{\mathcal{R}} = \lambda_z.$$

To this end, we show that if this was not the case, the formula complexity $\mathsf{L}(\mathcal{A}_{B4,Z} \times \mathcal{B}_{B4,Z})$ was at most $2^{2\sqrt{m}} \cdot m$ – thus contradicting the lower bound of $2^{\ell - \tilde{O}(\sqrt{m})}$ we have on $\mathsf{L}(\mathcal{A}_{B4,Z} \times \mathcal{B}_{B4,Z})$ (for an appropriate choice of the constant C in the definition of ℓ). The upper bound of $2^{2\sqrt{m}} \cdot m$ is derived by considering the following protocol for $KW_{\mathcal{A}_{B4,Z} \times \mathcal{B}_{B4,Z}}$: Alice sends to Bob $a|_{\mathcal{R}}$. By assumption, $a|_{\mathcal{R}} \neq b|_{\mathcal{R}}$, so now Bob knows a coordinate i such that $a_i \neq b_i$ and sends it to Alice. At this point, they solved $KW_{\mathcal{A}_{B4,Z} \times \mathcal{B}_{B4,Z}}$. It is not hard to see that the size of this protocol is at most $2^{2\sqrt{m}} \cdot m$. Hence, there exist a, b, λ_z as above.

Due to the step of "removing the small classes" on Alice's side, we know that the fraction of the strings $a' \in \mathcal{A}_{B4,Z}$ that satisfy $a'|_{\mathcal{R}} = \lambda_z$ is at least $2^{-3\cdot\sqrt{m}\cdot\log m}$: To see why, first observe that $\mathcal{A}_{B4} = \mathcal{A}_{A4} \subseteq \mathcal{A}_{A3}$. Then, recall that in the step of "removing the small classes", we partitioned $\mathcal{A}_{A3}$ to classes which were labeled by pairs $(\mathcal{R}', \lambda')$, and we obtained $\mathcal{A}_{A4}$ by removing the classes that consisted of less than $2^{-3\cdot\sqrt{m}\cdot\log m}$ fraction of the strings in $\mathcal{A}_{A3}$. Now, we know that there is a string $a \in \mathcal{A}_{B4}$ and r_Z such that $a|_{\mathcal{R}} = \lambda_Z$, and this implies that the class labeled by $(\mathcal{R}, \lambda_Z)$ was not removed. Hence, this class, consists of at least $2^{-3\cdot\sqrt{m}\cdot\log m}$ fraction of the strings in $\mathcal{A}_{A3}$, and in particular consists of at least $2^{-3\cdot\sqrt{m}\cdot\log m}$ fraction of the strings in $\mathcal{A}_{B4}$. A similar argument shows that at least $2^{-3\cdot\sqrt{m}}$ fraction of the strings $b' \in \mathcal{B}_{B4,Z}$ satisfy $b|_{\mathcal{R}} = \lambda_z$.

We now define for every Z the sets

$$\begin{aligned}\mathcal{A}_{\mathrm{agr},Z} &= \{a \in \mathcal{A}_{B4} : a|_{\mathcal{R}} = \lambda_Z\}\\ \mathcal{B}_{\mathrm{agr},Z} &= \{b \in \mathcal{B}_{B4} : b|_{\mathcal{R}} = \lambda_Z\}\end{aligned}$$

and remove all the edges of Z that correspond to strings outside $\mathcal{A}_{\mathrm{agr},Z}$ and $\mathcal{B}_{\mathrm{agr},Z}$. We denote the resulting graph by G_{agr}. We summarize the properties of G_{agr}:

- For every Z, it holds that $a|_{\mathcal{R}} = b|_{\mathcal{R}}$ for all $a \in \mathcal{A}_{\mathrm{agr},Z}$ and $b \in \mathcal{B}_{\mathrm{agr},Z}$.
- The graph G_{agr} retains at least $2^{-\tilde{O}(\sqrt{m})}$ fraction of the edges of G_{B4} on the $\mathcal{X}$-side, and hence at least $2^{-\tilde{O}(\sqrt{m})}$ fraction of the edges of G_{A3} on the $\mathcal{X}$-side. Similarly, G_{agr} contains at least $2^{-\tilde{O}(\sqrt{m})}$ fraction of the edges of G_{B3} on the $\mathcal{Y}$-side.
- The Z's are "roughly regular": For every Z, its degree on the $\mathcal{X}$-side is between $2^{-\tilde{O}(\sqrt{m})} \cdot d_{\mathcal{Z},\mathcal{X}}$ and $d_{\mathcal{Z},\mathcal{X}}$. The same holds for the $\mathcal{Y}$-side and $d_{\mathcal{Z},\mathcal{Y}}$.

7.6 Clean-up

We are almost ready to define our almost-hard distribution. Recall that this distribution is going to be defined by sampling a uniformly distributed path $X - Z - Y$ on a graph G', and that we denote by $j_1, \ldots, j_m$ the axes of the edges in Z. We would like this distribution to satisfy the following properties:

- For every $i \in \mathcal{R}$, it holds that $X_i = Y_i$ with probability 1.
- For every $i \in [m] - \mathcal{R}$ and every specific choice X^*, the min-entropy $H_\infty(j_i|X = X^*)$ is at least $\log n - \tilde{O}(\sqrt{m})$. The same holds for Y^*'s.

The first property holds for the distribution of G_{agr}. The second property basically follows from our step of "finding the revealed rows" in Alice's and Bob's sides, that is, Inequalities 11 and 12 above. However, the latter inequalities were proved for G_{A3} and G_{B3} respectively, and they do not imply similar inequalities for G_{agr} because of two issues:

- G_{agr} contains only some of the edges of G_{A3}, and this may cause the min-entropy $H(j_i|X^*)$ in G_{agr} to be much smaller than in G_{A3}.
  We note that this is an issue only for a minority of the vertices X^*: since G_{agr} retains $2^{-\tilde{O}(\sqrt{m})}$ fraction of the edges of G_{A3}, it holds that the degree of the average X^* is at least $2^{-\tilde{O}(\sqrt{m})}$ fraction of its degree in G_{A3}. For such vertices X^*, the min-entropy $H(j_i|X^*)$ is still sufficiently large. However, in order for the above second property to hold, we need the min-entropy to be large for *every* X^*. Similar considerations apply for Y^* and G_{B3}.
- When we proved the lower bound on the min-entropy $H_\infty(j_i|X^*)$ for G_{A3}, we assumed that Z is a uniformly distributed neighbor of X^*. However, in a uniformly distributed path $X - Z - Y$, this is not necessarily the case.
  It turns out that this is not a problem: It can be shown that the probability of each specific choice Z^* of Z is at most $2^{\tilde{O}(\sqrt{m})}$ times the probability of any other specific choice, and this is sufficiently good for our purposes. This follows from the "rough regularity" of the Z's, i.e., the fact that the degree of each specific choice Z^* on the $\mathcal{X}$-side is at most $2^{\tilde{O}(\sqrt{m})}$ larger than the degree of any other specific choice. The same argument works for the Y^*'s.

We could try to resolve the first issue by removing from G_{agr} the X's and Y's whose degree is too low. However, this might harm the rough regularity of the Z's, since it may cause some of the Z's to lose too many edges. We could fix the rough regularity by removing the Z's whose degree is too small, but then we will have X's and Y's with low degrees again. Fortunately, it turns out that if we repeat this process sufficiently many times, we end up with a graph in which all X's, Y's, and Z's have sufficiently large degrees. We choose the latter graph to be G'.

We turn to describing G' formally. Let $\varepsilon > 0$ be a number such that

- G_{agr} retains at least ε fraction of the edges of G_{A3} (respectively, G_{B3}) on the $\mathcal{X}$-side (respectively, on the $\mathcal{Y}$-side).
- Every Z in G_{agr} has degree at least $\varepsilon \cdot d_{\mathcal{Z},\mathcal{X}}$ (respectively, $\varepsilon \cdot d_{\mathcal{Z},\mathcal{Y}}$) on the $\mathcal{X}$-side (respectively, on the $\mathcal{Y}$-side).

It holds that $\varepsilon = 2^{-\tilde{O}(\sqrt{m})}$. We define the graph G' to be the graph obtained from G_{agr} by performing the following steps iteratively, until there are no more vertices to remove:

1. Remove all the vertices X whose degree is less than $\frac{1}{4} \cdot \varepsilon^3$ fraction of their degree in G_{A3}.
2. Remove all the vertices Y whose degree is less than $\frac{1}{4} \cdot \varepsilon^3$ fraction of their degree in G_{B3}.
3. Remove all the vertices Z whose degree on the $\mathcal{X}$-side is less than $\frac{1}{4} \cdot \varepsilon \cdot d_{\mathcal{Z},\mathcal{X}}$.
4. Remove all the vertices Z whose degree on the $\mathcal{Y}$-side is less than $\frac{1}{4} \cdot \varepsilon \cdot d_{\mathcal{Z},\mathcal{Y}}$.

When the process ends, we define the resulting graph to be G'. Our almost-hard distribution will be the distribution of G'. However, in order for this distribution to be well defined, we need to prove that G' is not empty. The basic idea of the proof is the following: First, we observe that Steps 1 and 2 cannot remove too many edges, since they only remove vertices whose degree is much lower than the average degree. Then, we observe that Steps 3 and 4 cannot remove too many Z's – the reason is that a vertex Z is only removed if many of its edges were removed in Steps 1 and 2. Finally, we observe that since only a few Z's are removed in Steps 3 and 4, and since the Z's are roughly regular, then those steps also cannot remove too many edges. We conclude that the process has not removed too many edges in all of the steps, and hence some edges must have remained. Details follow.

In order to prove that G' is not empty, we upper bound the number of edges that are removed by the foregoing process, and show that this number is less than the total number of edges of G_{agr}. First, we define some notation:

- We denote by $e_{A3,\mathcal{X}}$ and $e_{A3,\mathcal{Y}}$, the numbers of edges of G_{A3} on the $\mathcal{X}$-side and $\mathcal{Y}$-side respectively. We similarly denote $e_{B3,\mathcal{X}}$, $e_{B3,\mathcal{Y}}$, $e_{\text{agr},\mathcal{X}}$ and $e_{\text{agr},\mathcal{Y}}$ for G_B and G_{agr}.
- We denote $\mathcal{Z}_{\text{agr}}$ the set of Z's of G_{agr}.
- We denote by $\mathcal{X}$ and $\mathcal{Y}$ the sets of X's and Y's in G_{agr}. Observe that $\mathcal{X}$ is equal to the set of X's in G_{A3}, and $\mathcal{Y}$ is equal to the set of Y's in G_{B3}.
- For every $X \in \mathcal{X}$, we denote by d_X the degree of X in G_{A3}. Note that this is the degree in G_{A3}, and *may be different than the degree in G_{B3} or G_{agr}*.
- With some abuse of notation, for every $Y \in \mathcal{Y}$, we denote by d_Y the degree of Y in G_{B3}. *Note that this is the degree in G_{B3} and not in G_{A3}.*

We now prove that the $\mathcal{X}$-side of G' is not empty, and a similar proof holds for the $\mathcal{Y}$-side. To this end, we upper bound the number of edges on the $\mathcal{X}$-side that are removed in each step of the iterative construction above, and show that the total number of edges removed is less than $e_{\text{agr},\mathcal{X}}$. We start our proof by upper bounding the total number of edges that are removed in Step 1 above (in all iterations combined): Whenever we remove a vertex X, we remove at most $\frac{1}{4} \cdot \varepsilon^3 \cdot d_X$ edges. Hence, the total number of edges that are removed in Step 1 is at most

$$\sum_{X \in \mathcal{X}} \frac{1}{4} \cdot \varepsilon^3 \cdot d_X = \frac{1}{4} \cdot \varepsilon^3 \cdot \sum_{X \in \mathcal{X}} d_X = \frac{1}{4} \cdot \varepsilon^3 \cdot e_{A3,\mathcal{X}} \leq \frac{1}{4} \cdot \varepsilon^2 \cdot e_{\text{agr},\mathcal{X}}, \tag{13}$$

where the inequality holds since $e_{\text{agr},\mathcal{X}} \geq \varepsilon \cdot e_{A3,\mathcal{X}}$ by the definition of ε. Next, observe that the number of edges on the $\mathcal{X}$-side that are removed in Step 3 (in all iterations combined) is at most

$$\frac{1}{4} \cdot \varepsilon \cdot d_{\mathcal{Z},\mathcal{X}} \cdot |\mathcal{Z}_{\text{agr}}| \leq \frac{1}{4} \cdot e_{\text{agr},\mathcal{X}},$$

where the inequality follows from the fact that every $Z \in \mathcal{Z}_{\text{agr}}$ has at least $\varepsilon \cdot d_{\mathcal{Z},\mathcal{X}}$ edges on the $\mathcal{X}$-side in G_{agr}. Finally, we upper bound the number of edges that are removed on the $\mathcal{X}$-side in Step 4 (again, in all iterations combined): In order for a vertex Z to be removed in Step 4, we must have removed at least $\frac{3}{4} \cdot \varepsilon \cdot d_{\mathcal{Z},\mathcal{Y}}$ of its edges on the $\mathcal{Y}$-side previously. Those edges could only be removed in Step 2. On the other hand, it can be shown that the total number of edges removed on the $\mathcal{Y}$-side in Step 2 is at most $\frac{1}{4} \cdot \varepsilon^2 \cdot e_{\text{agr},\mathcal{Y}}$ using the same argument as in Inequality 13. Therefore the total number of $\mathcal{Z}$'s that are removed in Step 4 is at most

$$\frac{\frac{1}{4} \cdot \varepsilon^2 \cdot e_{\text{agr},\mathcal{Y}}}{\frac{3}{4} \cdot \varepsilon \cdot d_{\mathcal{Z},\mathcal{Y}}} \leq \frac{\frac{1}{4} \cdot \varepsilon^2 \cdot d_{\mathcal{Z},\mathcal{Y}} \cdot |\mathcal{Z}_{\text{agr}}|}{\frac{3}{4} \cdot \varepsilon \cdot d_{\mathcal{Z},\mathcal{Y}}} = \frac{1}{3} \cdot \varepsilon \cdot |\mathcal{Z}_{\text{agr}}| \,.$$

where the inequality again follows from the fact that every $Z \in \mathcal{Z}_{\text{agr}}$ has at least $\varepsilon \cdot d_{\mathcal{Z},\mathcal{Y}}$ edges on the $\mathcal{Y}$-side in G_{agr}. Now, note that each of those Z's can have at most $d_{\mathcal{Z},\mathcal{X}}$ edges on the $\mathcal{X}$-side, so the total number of edges that are removed in Step 4 on the $\mathcal{X}$-side is at most

$$\frac{1}{3} \cdot \varepsilon \cdot d_{\mathcal{Z},\mathcal{X}} \cdot |\mathcal{Z}_{\text{agr}}| \leq \frac{1}{3} \cdot e_{\text{agr},\mathcal{X}}.$$

Summing up, the total number of edges that are removed on the $\mathcal{X}$-side is at most

$$\frac{1}{4} \cdot \varepsilon^2 \cdot e_{\text{agr},\mathcal{X}} + \frac{1}{4} \cdot e_{\text{agr},\mathcal{X}} + \frac{1}{3} \cdot e_{\text{agr},\mathcal{X}} < e_{\text{agr},\mathcal{X}},$$

and therefore G' is non-empty on the $\mathcal{X}$-side. Similarly, it can be shown that G' is non-empty on the $\mathcal{Y}$-side, as required.

7.7 The almost-hard distribution

As mentioned above, our almost-hard distribution is the distribution of G': choose a uniformly distributed path $X - Z - Y$ in G', and output (X, Y). We now prove that this is indeed an $\tilde{O}(\sqrt{m})$-almost hard distribution. Clearly, for every $i \in \mathcal{R}$ it holds that $X_i = Y_i$ with probability 1. For every $i \in [m] - \mathcal{R}$ it either holds that $X_i = Y_i$ or it holds that X_i and Y_i disagree on exactly one coordinate, which is j_i, the i-th axis of Z. It remains to prove that for every X^* or Y^*, it holds that

$$H_\infty(j_i | X = X^*) \geq \log n - \tilde{O}(\sqrt{m}) \tag{14}$$

$$H_\infty(j_i | Y = Y^*) \geq \log n - \tilde{O}(\sqrt{m}). \tag{15}$$

We use the following claim, whose proof is deferred to the end of this section.

▶ **Claim 7.2.** *Fix a specific choice X^* of X. The probability of each specific choice Z^* of Z to be chosen conditioned on $X = X^*$ is at most $2^{\tilde{O}(\sqrt{m})}$ times larger than the probability of any other specific choice. The same holds for Y^*.*

We now prove Inequality 14, and Inequality 15 can be proved similarly. Basically, Inequality 14 follows from the corresponding inequality for G_{A3} (Inequality 11). As discussed in Section 7.6, there are two issues to deal with: First, the latter inequality assumes that Z is uniformly distributed, while in Inequality 14 the vertex Z is not uniformly distributed – this issue is resolved using Claim 7.2. Second, the degree of X^* in G' is smaller than its degree in G_{A3} – however, it is only smaller by a factor of $2^{\tilde{O}(\sqrt{m})}$, so this does not decrease the min-entropy of j_i by too much. We now provide the formal argument, which is a straightforward calculation.

Fix $i \in [m] - \mathcal{R}$, and fix a specific choice X^* of X. Fix a specific choice j^* for j_i, and let $\mathcal{Z}_{i,j^*}$ be the set of neighbors Z^* of X^* in G' whose axis on the i-th row is j^*. Recall that we proved that in G_{A3}, if Z is a *uniformly distributed* neighbor of X^*, then

$$H_\infty(j_i|X = X^*) \geq \log n - \tilde{O}(\sqrt{m}).$$

This implies in particular that under this distribution it holds that

$$\Pr[j_i = j^*|X = X^*] \leq \frac{2^{\tilde{O}(\sqrt{m})}}{n}.$$

In other words, this means that $\mathcal{Z}_{i,j^*}$ constitutes at most $2^{\tilde{O}(\sqrt{m})}/n$ fraction of the neighbors of X^* in G_{A3}. Next, observe that by our construction of G', the degree of X^* in G' is at least $2^{-\tilde{O}(\sqrt{m})}$ fraction of its degree in G_{A3}. Therefore $\mathcal{Z}_{i,j^*}$ constitutes at most $2^{\tilde{O}(\sqrt{m})}/n$ fraction of *the neighbors of* X^* *in* G'. Finally, the latter fact together with Claim 7.2 implies that the probability that $Z \in \mathcal{Z}_{i,j^*}$ is at most $2^{\tilde{O}(\sqrt{m})}/n$, as required. This concludes the proof of the main lemma.

Proof of Claim 7.2. Fix choices X^* and Z^*. For every specific choice Z' of Z, it holds that

$$\frac{\Pr[Z^*|X^*]}{\Pr[Z'|X^*]} = \frac{\Pr[Z^* \text{ and } X^*]}{\Pr[Z' \text{ and } X^*]}.$$

Now, $\Pr[Z^* \text{ and } X^*]$ is the probability of the edge (X^*, Z^*) to be selected, which is proportional to the number of paths $X - Z - Y$ in which it participates. The latter number is exactly the degree of Z^* on the $\mathcal{Y}$-side, which is between $2^{-\tilde{O}(\sqrt{m})} \cdot d_{\mathcal{Z},\mathcal{Y}}$ and $d_{\mathcal{Z},\mathcal{Y}}$. The same holds for the probability $\Pr[Z' \text{ and } X^*]$. It thus follows that

$$\frac{\Pr[Z^* \text{ and } X^*]}{\Pr[Z' \text{ and } X^*]} \leq \frac{d_{\mathcal{Z},\mathcal{Y}}}{2^{-\tilde{O}(\sqrt{m})} \cdot d_{\mathcal{Z},\mathcal{Y}}} \leq 2^{\tilde{O}(\sqrt{m})},$$

as required. ◀

8 Average-Case Lower Bounds

In this section, we prove average-case analogues of our main theorem (in Section 8.1) and of the cubic lower bound for Andreev's function (in Section 8.2). Hardness on-average is defined as follows.

▶ **Definition 8.1.** A function $F : \{0,1\}^N \to \{0,1\}$ is said to be (s, ε)-*hard* if every formula of size at most s computes F correctly on at most $\frac{1}{2} + \varepsilon$ fraction of the inputs.

8.1 Average-case lower bound for composition

We prove the following theorem, which is an average-case analogue of our main theorem.

▶ **Theorem 1.3** (restated). *Let $f : \{0,1\}^m \to \{0,1\}$ be an (s, ε)-hard function. Then, $f \diamond \oplus_n$ is $(s', \varepsilon + 2^{-m})$-hard for*

$$s' \geq s \cdot \mathsf{L}(\oplus_n)/2^{\tilde{O}(\sqrt{m+\log n})}.$$

To this end, we use the following immediate corollary of the Karchmer-Wigderson connection (Theorem 2.11).

▶ **Corollary 8.2.** *A function $F : \{0,1\}^N \to \{0,1\}$ is (s,ε)-hard if and only if for every two sets $\mathcal{X} \subseteq F^{-1}(0)$ and $\mathcal{Y} \subseteq F^{-1}(1)$ such that $|\mathcal{X}| + |\mathcal{Y}| > (\frac{1}{2} + \varepsilon) \cdot 2^N$, it holds that $\mathsf{L}(KW_{\mathcal{X}\times\mathcal{Y}}) \geq s$.*

Let $f : \{0,1\}^m \to \{0,1\}$ be an (s,ε)-hard function and let $\mathcal{X} \subseteq (f \diamond \oplus_n)^{-1}(0)$ and $\mathcal{Y} \subseteq (f \diamond \oplus_n)^{-1}(1)$ be such that $(\frac{1}{2} + \varepsilon + 2^{-m}) \cdot 2^{m \cdot n}$. Our goal is to prove that

$$\mathsf{L}(KW_{\mathcal{X}\times\mathcal{Y}}) \geq s \cdot n^2 / 2^{\tilde{O}(\sqrt{m+\log n})}.$$

In order to do so, we prove that the rectangle $\mathcal{X}\times\mathcal{Y}$ satisfies the requirement of the generalized f-stage lemma (Lemma 4.5). We then derive the lower bound by plugging the latter lemma into the proof of the main theorem in Section 3.3.

Recall that for every product of edges $Z = (Z^0, Z^1)$, we define the f-rectangle of $\mathcal{X} \times \mathcal{Y}$ with respect to Z as the rectangle $\mathcal{A}_Z \times \mathcal{B}_Z$ where

$$\begin{aligned} \mathcal{A}_Z &\stackrel{\text{def}}{=} \left\{a \in f^{-1}(0) | Z^a \in \mathcal{X}\right\} \\ B_Z &\stackrel{\text{def}}{=} \left\{b \in f^{-1}(1) | Z^b \in \mathcal{Y}\right\}. \end{aligned}$$

In order to show that the rectangle $\mathcal{X} \times \mathcal{Y}$ satisfies the requirement of the generalized f-stage lemma, we need to show that for at least 2^{-m} fraction of the Z's it holds that $\mathsf{L}(KW_{\mathcal{A}_Z\times\mathcal{B}_Z}) \geq s$. To this end, it suffices to prove that at least 2^{-m} fraction of the Z's satisfy that $|\mathcal{A}_Z| + |\mathcal{B}_Z| \geq (\frac{1}{2} + \varepsilon) \cdot 2^m$, and this will imply the required lower bound on $\mathsf{L}(KW_{\mathcal{A}_Z\times\mathcal{B}_Z})$ by the average-case hardness of f. We prove this via a straightforward averaging argument.

More specifically, consider the following bipartite graph G: One side of the graph is the set $\mathcal{X} \cup \mathcal{Y}$, and other side is the set $\mathcal{Z}$ of all Z's. A matrix $W \in \mathcal{X} \cup \mathcal{Y}$ is connected to $Z \in \mathcal{Z}$ if and only if $W = Z^w$ for some $w \in \{0,1\}^m$. It is easy to see that the degree of every $W \in \mathcal{X} \cup \mathcal{Y}$ is exactly n^m, so the total number of edges in the graph is

$$|\mathcal{X} \cup \mathcal{Y}| \cdot n^m \geq (\frac{1}{2} + \varepsilon + 2^{-m}) \cdot 2^{m \cdot n} \cdot n^m = (\frac{1}{2} + \varepsilon + 2^{-m}) \cdot 2^m \cdot |\mathcal{Z}|,$$

where the equality holds since $|\mathcal{Z}| = 2^{m\cdot(n-1)} \cdot n^m$. On the other hand, the degree of each Z in this graph is exactly $|\mathcal{A}_Z| + |\mathcal{B}_Z|$. Now, the Z's whose degree is less than $\left(\frac{1}{2} + \varepsilon\right) \cdot 2^m$ contribute less than

$$\left(\frac{1}{2} + \varepsilon\right) \cdot 2^m \cdot |\mathcal{Z}|$$

edges. Therefore, at least $2^{-m} \cdot 2^m \cdot |\mathcal{Z}|$ edges are connected to Z's whose degree is at least $\left(\frac{1}{2} + \varepsilon\right) \cdot 2^m$. The degree of every such Z is at most 2^m, and therefore the number of such Z's must be at least:

$$2^{-m} \cdot 2^m \cdot |\mathcal{Z}| / 2^m = 2^{-m} \cdot |\mathcal{Z}|.$$

It thus follows that $|\mathcal{A}_Z| + |\mathcal{B}_Z| \geq (\frac{1}{2} + \varepsilon) \cdot 2^m$ for at least 2^{-m} fraction of the Z's, and therefore the rectangle $\mathcal{X} \times \mathcal{Y}$ satisfies the requirement of the generalized f-stage lemma.

We finally turn to prove the lower bound. Fix a protocol Π that solves $KW_{\mathcal{X}\times\mathcal{Y}}$, and let us denote its size by S. Without loss of generality, we may assume that $S \leq 2^m \cdot n^2$, or otherwise we are done. We apply Theorem [6] 2.4 to Π with $\alpha = \frac{1}{\sqrt{m+\log n}}$, thus obtaining a

[6] See also the restatement of this theorem in Section 3.3

new protocol Π' of depth at most $2^{\tilde{O}(\sqrt{m+\log n})}$ and size $S' \leq S^{1+\frac{1}{\sqrt{m+\log n}}}$. We prove that $S' \geq s \cdot \mathsf{L}(\oplus_n)/2^{\tilde{O}(\sqrt{m+\log n})}$ and this will imply the same lower bound for S, as required (see Section 3.3 for details).

By the generalized f-stage lemma (Lemma 4.5), it follows that Π' has at least $s/2^{\tilde{O}(\sqrt{m+\log n})}$ partial transcripts π_1 that are alive, where none of them is an ancestor of another. By the structure theorem (Theorem 3.4), for each such partial transcript π_1 there are at least $\mathsf{L}(\oplus_n)/2^{\tilde{O}(\sqrt{m})}$ suffixes π_2 such that $\pi_1 \circ \pi_2$. Summing over all the possible choices for π_1 and π_2, it follows that Π' has at least $s \cdot \mathsf{L}(\oplus_n)/2^{\tilde{O}(\sqrt{m+\log n})}$ distinct transcripts, which is what we wanted to prove.

8.2 Average-case cubic lower bound

In the rest of this section, we prove Corollary 1.4, which gives average-case cubic lower bounds for a variant of the Andreev function due to Komargodski and Raz [23]. Our proof is essentially the same as that of [23], modulo the proof of Theorem 1.3, and some different choices of the parameters.

▶ **Corollary 1.4** (restated). *For every $n, c \in \mathbb{N}$ there exists a function $F_{n,c} : \{0,1\}^n \to \{0,1\}$ bits that is (S, n^{-c})-hard for*

$$S \geq n^{3-\tilde{O}(\frac{1}{\sqrt{\log n}})}.$$

Let $n, c \in \mathbb{N}$ be as in the theorem. Let $m \stackrel{\text{def}}{=} 10 \cdot c \cdot \log n$. Let $C : \{0,1\}^{n/2} \to \{0,1\}^{2^m}$ be the list-decodable code of Fact 2.22, and recall that the list-decodability means that for every string $w \in \{0,1\}^{2^m}$, there are at most 2^m codewords of C that are $(\frac{1}{2} - \frac{1}{2} \cdot \sqrt{\frac{n}{2^{m/2}}})$-close to w. The function $F_{n,c}$ is defined as follows: The input of $F_{n,c}$ consists of two parts, each of length $n/2$. The first part of the input is denoted f. Recall that $C(f)$ is a string of length 2^m, and we view it as a truth table of a function from $\{0,1\}^m$ to $\{0,1\}$. The second part of the input is a sequence $x_1, \ldots, x_m$ of strings in $\{0,1\}^{n/2m}$. The function $F_{n,c}$ is now defined by

$$F_{n,c}(f, x_1, \ldots, x_m) \stackrel{\text{def}}{=} \left(C(f) \diamond \oplus_{\frac{n}{2m}}\right)(x_1, \ldots, x_m).$$

We use the following claim, which is proved by a straightforward counting argument.

▶ **Claim 8.3.** *Let f be a uniformly distributed string in $\{0,1\}^{n/2}$. Then, the function $C(f)$ is (s, n^{-2c})-hard for $s \stackrel{\text{def}}{=} n/16 \cdot \log m$ with probability at least $1 - 2^{-n/5}$.*

Proof. We count the number of functions from $\{0,1\}^m$ to $\{0,1\}$ that can be approximated by formulas of size $s \stackrel{\text{def}}{=} n/16 \cdot \log m$. Following a calculation in [17] (see the proof of Theorem 1.23), the number of formulas of size s over m variables is at most $(9m)^s \leq 2^{n/4}$. By the list-decodability of C, for each such formula ϕ there are at most 2^m strings $h \in \{0,1\}^{n/2}$ such that

$$\Pr_{x \leftarrow \{0,1\}^{n/2}}[C(h)(x) = \phi(x)] > \frac{1}{2} + n^{-2c} \geq \frac{1}{2} + \frac{1}{2} \cdot \sqrt{\frac{n}{2^{m/2}}}. \tag{16}$$

It follows that the total number of strings h that satisfy Inequality 16 for *any* formula of size s is at most $2^{n/4} \cdot 2^m$. Therefore, if f is chosen uniformly at random, the probability that $C(f)$ is (s, n^{-2c})-hard is at least

$$1 - \frac{2^{n/4} \cdot 2^m}{2^{n/2}} \geq 1 - 2^{-n/5},$$

as required. ◀

By Theorem 1.3, for every fixed choice of f for which $C(f)$ is (s, n^{-2c})-hard, it holds that $C(f) \diamond \oplus_{\frac{n}{2m}}$ is $(S, n^{-2c} + 2^{-m})$-hard for

$$S \stackrel{\text{def}}{=} s \cdot n^2 / 2^{\tilde{O}(\sqrt{m+\log n})} = n^{3-\tilde{O}(\frac{1}{\sqrt{\log n}})}.$$

Therefore, for every such fixed choice of f and every fixed formula ϕ of size S, it holds that

$$\Pr_{x_1,\ldots,x_m \leftarrow \{0,1\}^{n/2m}} [F_{n,c}(f, x_1, \ldots, x_m) = \phi(f, x_1, \ldots, x_m)] \leq \frac{1}{2} + n^{-2c} + 2^{-m}.$$

Now, let f be uniformly distributed, and let H_f denote the event in which $C(f)$ is (s, n^{-2c})-hard. It follows that for every formula ϕ of size at most S and for uniformly distributed f and $x_1, \ldots, x_m$ it holds that

$$\begin{aligned}
& \Pr [F_{n,c}(f, x_1, \ldots, x_m) = \phi(f, x_1, \ldots, x_m)] \\
\leq\ & \Pr [F_{n,c}(f, x_1, \ldots, x_m) = \phi(f, x_1, \ldots, x_m) | H_f] + \Pr [\neg H_f] \\
\leq\ & \frac{1}{2} + n^{-2c} + 2^{-m} + 2^{-n/5} \\
\leq\ & \frac{1}{2} + n^{-c}.
\end{aligned}$$

Hence, $F_{n,c}$ is (s, n^{-c})-hard for $S \geq n^{3-\tilde{O}(\frac{1}{\sqrt{\log n}})}$, as required.

9 Future Directions and Open Problems

In order to prove the KRW conjecture, one should replace the parity function in our result with a general function $g : \{0,1\}^n \to \{0,1\}$. It seems to us that a good starting point would be to prove the KRW conjecture for the composition of a universal relation and a function g, denoted $U \diamond g$. We now explain what this composition is, and then discuss how one might prove the KRW composition for it.

The composition $U \diamond g$

The *universal relation* is the following communication problem: Alice and Bob get two distinct strings $x, y \in \{0,1\}^m$, and should find a coordinate on which x and y disagree. The difference between the universal relation and KW relations is that x and y are not required to be a 0-preimage and 1-preimage of some function f. This makes the universal relation much simpler and easier to analyze, and therefore the universal relation is often a good starting point for studying KW relations. For convenience, we denote the universal relation by U.

As was observed by [15], it is often useful to relax the requirement that x and y are distinct as follows: We allow x and y to be equal, but in this case, we also allow Alice and Bob to reject the inputs instead of outputting a coordinate. It is not hard to show that this relaxation does not increase the complexity of the problem by much. It is well-known that the communication complexity of the (relaxed) universal relation is at least m, and that the "hardest inputs" are those in which $x = y$ [20, 10, 15, 12].

The composition $U \diamond g$ is the following communication problem: Alice and Bob get as inputs $m \times n$ matrices X and Y respectively such that $g(X) \neq g(Y)$, and their goal is to find an entry (i, j) such that $X_{i,j} \neq Y_{i,j}$. Again, we relax the requirement that $g(X) \neq g(Y)$ as follows: We allow X and Y to satisfy $g(X) = g(Y)$, but in this case, we also allow Alice and Bob to reject the inputs and not output an entry (i, j). Here, too, the relaxation does not increase the complexity of the problem by much.

The KRW conjecture for $U \diamond g$

The analogue of the KRW conjecture for $U \diamond g$ would be to prove that

$$\mathsf{C}(U \diamond g) \approx \mathsf{C}(U) + \mathsf{C}(KW_g) \approx m + \mathsf{C}(KW_g)$$

(for simplicity, we focus on the communication complexity rather than on the protocol size). We could try to to prove it using the approach of this paper as follows. Suppose that there is a protocol Π that solves $U \diamond g$. Then, we would have liked to prove the following claims:

- **An analogue of the f-stage lemma:** There is a partial transcript of π_1 of length $m - \tilde{O}(\sqrt{m})$ that is alive, i.e., that has not solved the universal relation on $g(X)$ and $g(Y)$.
- **An analogue of the structure theorem:** Any live partial transcript π_1 has a suffix of length $\mathsf{C}(KW_g) - \tilde{O}(\sqrt{m})$.

If we could prove those two claims, they would have implied the lower bound

$$\mathsf{C}(U \diamond g) \geq m + \mathsf{C}(KW_g) - \tilde{O}(\sqrt{m}), \tag{17}$$

which would have been sufficiently good for our purposes.

An analogue of the f-stage lemma

Recall that in Section 3, we implemented the above approach by defining products of edges $Z = (Z^0, Z^1)$. We then invoked the protocol on inputs X and Y of the form $X = Z^a$, $Y = Z^b$ for $a \in f^{-1}(0)$ and $b \in f^{-1}(1)$. In particular, we proved the f-stage lemma by considering the invocation of the protocol on such inputs for different Z's.

We would like to prove an analogue of the f-stage lemma for $U \diamond g$ using a similar strategy. To this end, we would like to invoke the protocol Π on inputs of the form $X = Z^a$, $Y = Z^b$, and show that it cannot solve the universal relation on a and b using $m - \tilde{O}(\sqrt{m})$ bits. A natural way to do so would be to choose the pair (a, b) to be a hard input for the universal relation.

As we noted above, the hard inputs to the universal relation are those in which $a = b$. Now, observe that whenever $a = b$, it also holds that $X = Y$. Thus, it seems that for an analogue of the f-stage lemma for $U \diamond g$, we should invoke the protocol Π on inputs of the form (X, X). This leads to the following natural definition for what it means that "π_1 is alive".

▶ **Definition 9.1.** We say that a partial transcript π_1 *is alive* if for at least $2^{-(m-\tilde{O}(\sqrt{m}))}$ fraction of the matrices $X \in \{0,1\}^{m \times n}$, the input (X, X) is consistent with π_1. In other words, if we denote by $\mathcal{X}_{\pi_1} \times \mathcal{Y}_{\pi_1}$ the rectangle of π_1, then $X \in \mathcal{X}_{\pi_1} \cap \mathcal{Y}_{\pi_1}$ for at least $2^{-(m-\tilde{O}(\sqrt{m}))}$ fraction of the matrices $X \in \{0,1\}^{m \times n}$.

Intuitively, this definition says that π_1 has gives at most $m - \tilde{O}(\sqrt{m})$ bits of information about the inputs of the players. In particular, π_1 gives at most $m - \tilde{O}(\sqrt{m})$ bits about a and b, and therefore it is still far from solving the universal relation on a and b. This intuition can be formalized using the ideas of [10, 15, 12], but it is not necessary for our discussion. The following analogue of the f-stage lemma can now be proved using a straightforward averaging argument.

▶ **Lemma 9.2** (Universal-stage lemma). *There is a live partial transcript π_1 of length $m - \tilde{O}(\sqrt{m})$ that has not solved the universal relation.*

An analogue of the structure theorem

The difficult part in proving the lower bound on $\mathsf{C}(U \diamond g)$ would be proving an analogue of the structure theorem. Such an analogue would say that if Alice and Bob have not solved the universal relation yet, then they must transmit $\mathsf{C}(KW_g) - \tilde{O}(\sqrt{m})$ more bits. Given Definition 9.1, this can be formalized as follows.

▶ **Conjecture 9.3.** *Let $\mathcal{X} \subseteq \{0,1\}^{m \times n}$ be a set of matrices of density at least $2^{-(m-\tilde{O}(\sqrt{m}))}$. Then, the restriction of $U \diamond g$ to the rectangle $\mathcal{X} \times \mathcal{X}$ has communication complexity at least $\mathsf{C}(KW_g) - \tilde{O}(\sqrt{m})$.*

We note that it is possible to construct artificial examples of functions g for which Conjecture 9.3 does not hold: in particular, if g is easy on $(1-\varepsilon)$-fraction of its inputs, it is possible that all the matrices in $\mathcal{X}$ contain only easy inputs as rows[7]. However, it might be possible to prove it for some "reasonable" class of functions, and that might be sufficient for proving formula lower bounds. For example, it might be possible to prove this conjecture for the case where g is a random function. We also note there is a simple (but non-trivial) proof of the conjecture for the case where $g = \oplus_n$ – in fact, this observation was the trigger to this work.

Another way to deal with the aforementioned artificial examples is to change the conjecture such that it allows us to get rid of the easy inputs of g. This is done by replacing $\{0,1\}^{m \times n}$ with some subset $\mathcal{X}_0$ that depends on g and should consist of the hard inputs:

▶ **Conjecture 9.4.** *For every non-constant function $g : \{0,1\}^n \to \{0,1\}$ there exists $\mathcal{X}_0 \subseteq \{0,1\}^{m \times n}$ such that the following holds: Let $\mathcal{X} \subseteq \mathcal{X}_0$ be a set of matrices of density at least $2^{-(m-\tilde{O}(\sqrt{m}))}$ in $\mathcal{X}_0$. Then, the restriction of $U \diamond g$ to the rectangle $\mathcal{X} \times \mathcal{X}$ has communication complexity at least $\mathsf{C}(KW_g) - \tilde{O}(\sqrt{m})$.*

It is not hard to see that Conjecture 9.4 is sufficient for proving the lower bound on $\mathsf{C}(U \diamond g)$: this can be done by replacing $\{0,1\}^{m \times n}$ with $\mathcal{X}_0$ in Definition 9.1 and Lemma 9.2 above.

Conjecture 9.4 could serve as the next intermediate goal toward proving the KRW conjecture, and we suggest it as an open problem. In fact, we do not know how to prove this conjecture even if the density of $\mathcal{X}$ in $\mathcal{X}_0$ is allowed to be as high as $\frac{1}{2}$, and the desired lower bound is allowed to be as small as $\mathsf{C}(KW_g) - 0.99 \cdot m$.

The 1-out-of-k problem

We now discuss a special case of Conjecture 9.3 which seems to be interesting in its own right. First, we define the following communication problem.

▶ **Definition 9.5** (The 1-out-of-k problem)**.** Let $g : \{0,1\}^n \to \{0,1\}$ be a non-constant function, and let $k \in \mathbb{N}$. The 1-*out-of-k version of* KW_g is the following communication problem: Alice and Bob get matrices $X, Y \in \{0,1\}^{k \times n}$ respectively such that

- $g(X)$ and $g(Y)$ are the all-zeroes and all-ones strings respectively.
- All the rows of X and Y are all distinct.

The goal of Alice and Bob is to find an entry (i,j) such that $X_{i,j} \neq Y_{i,j}$.

[7] Consider a function $g : \{0,1\}^n \to \{0,1\}$ that is defined as follows: given an input x, if the first five bits of x are all zeroes, then $g(x)$ is some hard function of the remaining bits, and otherwise $g(x) = x_6$. Now, consider the set $\mathcal{X} \subseteq \{0,1\}^{m \times n}$ that consists of all the matrices X in which there is no row with the first five bits all being zeroes. It is not hard to see that the communication complexity of $U \diamond g$ restricted to $\mathcal{X} \times \mathcal{X}$ is at most $m + O(1)$, and this might be much smaller than $\mathsf{C}(KW_g)$ if $m \ll n$.

Clearly, the communication complexity of the 1-out-of-k version of KW_g is at most $\mathsf{C}(KW_g)$, since Alice and Bob can run the optimal protocol for KW_g on the first rows of X and Y. The question is whether the communication complexity of the 1-out-of-k version of KW_g can be much smaller? We suggest proving the following conjecture as another open problem.

▶ **Conjecture 9.6.** *For every non-constant $g : \{0,1\}^n \to \{0,1\}$ and $k \in \mathbb{N}$, the communication complexity of the 1-out-of-k version of KW_g is at least $\mathsf{C}(KW_g) - \operatorname{poly}\log n$.*

Observe that Conjecture 9.6 is indeed a special case of Conjecture 9.3: the reason is that we can always choose the subset $\mathcal{X}$ to be the set of matrices X such that the first k bits of $g(X)$ are equal, and all the rows of X are distinct. The density of this set $\mathcal{X}$ is slightly less than 2^{-k}, and the communication complexity of the restriction of $U \diamond g$ to the rectangle $\mathcal{X} \times \mathcal{X}$ is at most the communication complexity of the 1-out-of-k version of KW_g.

We note that although we defined the 1-out-of-k problem only for KW relations, it could be generalized to other models of computation. For those models, one could state analogues of Conjecture 9.6 that are interesting in their own right. For example, consider the following analogues for communication complexity and circuit complexity:

▶ **Conjecture 9.7** (The 1-out-of-k problem for communication complexity)**.** *Let $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$, and consider the communication problem of computing f. The 1-out-of-k version of f is defined as follows: Alice gets distinct $x_1, \ldots, x_k \in \{0,1\}^n$, Bob gets distinct $y_1, \ldots, y_k \in \{0,1\}^n$, and their goal is to output an index $i \in [k]$ and the bit $f(x_i, y_i)$. The conjecture is that the communication complexity of this problem is at least $\mathsf{C}(f) - \operatorname{poly}\log n$.*

▶ **Conjecture 9.8** (The 1-out-of-k problem for circuit complexity)**.** *Let $f : \{0,1\}^n \to \{0,1\}$, and consider the problem of computing f using a boolean circuit. The 1-out-of-k version of f is defined as follows: a circuit gets as input distinct $x_1, \ldots, x_k$, and it should output an index $i \in [k]$ and the bit $f(x_i)$. The conjecture is that the circuit complexity of this problem is at least the circuit complexity of f up to a polynomial factor.*

The 1-out-of-k problem is a close variant of the "choose" problem introduced by Beimel et al. [3], who also posed conjectures that correspond to Conjectures 9.7 and 9.8. The difference between the 1-out-of-k problem defined above and the "choose" problem of [3] is that in the "choose" problem, the inputs are not required to be distinct, and on the other hand, we have k functions $f_1, \ldots, f_k$ instead of a single function f. The question is whether choosing one of the functions f_i and computing it on its corresponding input is easier than computing the easiest function among $f_1, \ldots, f_k$ in isolation.

[3] made an interesting observation, which also translates to the 1-out-of-k problem as follows: 1-out-of-k conjectures of the above form are implied by *direct-sum conjectures*. For concreteness, we explain this claim for the example of communication complexity. A direct-sum conjecture for communication complexity says that the complexity of computing k independent instances of f is $k \cdot \mathsf{C}(f)$. The observation of [3] is that the latter direct-sum conjecture implies that the communication complexity of the 1-out-of-k version of f is $\mathsf{C}(f)$.

To see why this is true, suppose there was a protocol that solved the 1-out-of-k version of f using less than $\mathsf{C}(f)$ bits. If this was the case, it would have been possible to compute k independent instances of f using less than $k \cdot \mathsf{C}(f)$ as follows: Alice and Bob first use the protocol for the 1-out-of-k version of f on the k instances, thus computing f on one instance. Then, they would compute f independently on each of the remaining instances. The complexity of this protocol would be $(k-1) \cdot \mathsf{C}(f)$ plus the complexity of the 1-out-of-k version of f, which is less than $k \cdot \mathsf{C}(f)$ by assumption.

Direct-sum conjectures have been studied in many different areas. In particular, the direct-sum conjecture for communication complexity has been proposed in [20], and partial results were obtained in [11, 19, 2, 6, 5] . In particular, the result of [11] implies that the complexity of the 1-out-of-k problem of Conjecture 9.7 above is at least $\sqrt{\mathsf{C}(f)}$. Unfortunately, the known results are insufficient for proving Conjecture 9.6. It is interesting question whether proving Conjectures 9.6 and 9.7 is easier than proving the corresponding direct-sum conjectures, or alternatively, whether 1-out-of-k conjectures imply direct-sum conjectures.

Acknowledgements. We would like to thank Avi Wigderson, Avishay Tal, Gillat Kol, Oded Goldreich, and Ilan Komargodski for useful discussions and ideas. We would also like to thank anonymous referees for comments that improved the presentation of this work.

References

1 Alexander E. Andreev. On a method for obtaining more than quadratic effective lower bounds for the complexity of π-schemes. *Moscow University Mathematics Bulletin*, 42(1):24–29, 1987.

2 Boaz Barak, Mark Braverman, Xi Chen, and Anup Rao. How to compress interactive communication. In *STOC*, pages 67–76, 2010.

3 Amos Beimel, Sebastian Ben Daniel, Eyal Kushilevitz, and Enav Weinreb. Choosing, agreeing, and eliminating in communication complexity. *Computational Complexity*, 23(1):1–42, 2014.

4 Maria Luisa Bonet and Samuel R. Buss. Size-depth tradeoffs for boolean formulae. *Inf. Process. Lett.*, 49(3):151–155, 1994.

5 Mark Braverman. Interactive information complexity. In *STOC*, pages 505–524, 2012.

6 Mark Braverman and Anup Rao. Information equals amortized communication. In *FOCS*, pages 748–757, 2011.

7 Richard P. Brent. The parallel evaluation of general arithmetic expressions. *J. ACM*, 21(2):201–206, 1974.

8 Ruiwen Chen, Valentine Kabanets, Antonina Kolokolova, Ronen Shaltiel, and David Zuckerman. Mining circuit lower bound proofs for meta-algorithms. *Computational Complexity*, 24(2):333–392, 2015.

9 Thomas M. Cover and Joy A. Thomas. *Elements of information theory*. Wiley-Interscience, 1991.

10 Jeff Edmonds, Russell Impagliazzo, Steven Rudich, and Jiri Sgall. Communication complexity towards lower bounds on circuit depth. *Computational Complexity*, 10(3):210–246, 2001.

11 Tomás Feder, Eyal Kushilevitz, Moni Naor, and Noam Nisan. Amortized communication complexity. *SIAM J. Comput.*, 24(4):736–750, 1995.

12 Dmitry Gavinsky, Or Meir, Omri Weinstein, and Avi Wigderson. Toward better formula lower bounds: an information complexity approach to the KRW composition conjecture. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 – June 03, 2014*, pages 213–222, 2014.

13 Michelangelo Grigni and Michael Sipser. Monotone separation of Logspace from NC. In *Structure in Complexity Theory Conference*, pages 294–298, 1991.

14 Johan Håstad. The shrinkage exponent of de Morgan formulas is 2. *SIAM J. Comput.*, 27(1):48–64, 1998.

15 Johan Håstad and Avi Wigderson. Composition of the universal relation. In *Advances in computational complexity theory, AMS-DIMACS*, 1993.

16 Russell Impagliazzo and Noam Nisan. The effect of random restrictions on formula size. *Random Struct. Algorithms*, 4(2):121–134, 1993.

17 Stasys Jukna. *Boolean Function Complexity – Advances and Frontiers*, volume 27 of *Algorithms and combinatorics*. Springer, 2012.

18 Yael Tauman Kalai and Ran Raz. Interactive PCP. In *ICALP (2)*, pages 536–547, 2008.

19 Mauricio Karchmer, Eyal Kushilevitz, and Noam Nisan. Fractional covers and communication complexity. *SIAM J. Discrete Math.*, 8(1):76–92, 1995.

20 Mauricio Karchmer, Ran Raz, and Avi Wigderson. Super-logarithmic depth lower bounds via the direct sum in communication complexity. *Computational Complexity*, 5(3/4):191–204, 1995.

21 Mauricio Karchmer and Avi Wigderson. Monotone circuits for connectivity require super-logarithmic depth. *SIAM J. Discrete Math.*, 3(2):255–265, 1990.

22 V. M. Khrapchenko. A method of obtaining lower bounds for the complexity of π-schemes. *Mathematical Notes Academy of Sciences USSR*, 10:474–479, 1972.

23 Ilan Komargodski and Ran Raz. Average-case lower bounds for formula size. In *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 171–180, 2013.

24 Ilan Komargodski, Ran Raz, and Avishay Tal. Improved average-case lower bounds for de Morgan formula size. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 588–597, 2013.

25 Eyal Kushilevitz and Noam Nisan. *Communication complexity*. Cambridge University Press, 1997.

26 Dana Moshkovitz. Parallel repetition from fortification. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 414–423, 2014.

27 Mike Paterson and Uri Zwick. Shrinkage of de Morgan formulae under restriction. *Random Struct. Algorithms*, 4(2):135–150, 1993.

28 Ran Raz and Avi Wigderson. Monotone circuits for matching require linear depth. *J. ACM*, 39(3):736–744, 1992.

29 Alexander A. Razborov. Applications of matrix methods to the theory of lower bounds in computational complexity. *Combinatorica*, 10(1):81–93, 1990.

30 Rahul Santhanam. Fighting perebor: New and improved algorithms for formula and QBF satisfiability. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*, pages 183–192, 2010.

31 Philip M. Spira. On time-hardware complexity tradeoffs for boolean functions. In *Proceedings of the Fourth Hawaii International Symposium on System Sciences*, pages 525–527, 1971.

32 Bella Abramovna Subbotovskaya. Realizations of linear functions by formulas using +,.,-. *Soviet Mathematics Doklady*, 2:110–112, 1961.

33 Madhu Sudan. Algorithmic introduction to coding theory (lecture notes), 2001. Available from `http://theory.csail.mit.edu/~madhu/FT01/`.

34 Avishay Tal. Shrinkage of de Morgan formulae by spectral techniques. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 551–560, 2014.

35 Andrew Chi-Chih Yao. Some complexity questions related to distributive computing (preliminary report). In *STOC 1979*, pages 209–213, 1979.

Nearly Optimal Separations Between Communication (or Query) Complexity and Partitions

Andris Ambainis[1], Martins Kokainis[2], and Robin Kothari[3]

1 **Faculty of Computing, University of Latvia, Riga, Latvia**
`andris.ambainis@lu.lv`
2 **Faculty of Computing, University of Latvia, Riga, Latvia**
`martins.kokainis@lu.lv`
3 **Center for Theoretical Physics, Massachusetts Institute of Technology, Cambridge, USA**
`rkothari@mit.edu`

Abstract

We show a nearly quadratic separation between deterministic communication complexity and the logarithm of the partition number, which is essentially optimal. This improves upon a recent power 1.5 separation of Göös, Pitassi, and Watson (FOCS 2015). In query complexity, we establish a nearly quadratic separation between deterministic (and even randomized) query complexity and subcube partition complexity, which is also essentially optimal. We also establish a nearly power 1.5 separation between quantum query complexity and subcube partition complexity, the first superlinear separation between the two measures. Lastly, we show a quadratic separation between quantum query complexity and one-sided subcube partition complexity.

Our query complexity separations use the recent cheat sheet framework of Aaronson, Ben-David, and Kothari. Our query functions are built up in stages by alternating function composition with the cheat sheet construction. The communication complexity separation follows from "lifting" the query separation to communication complexity.

1998 ACM Subject Classification F1.2 Modes of Computation

Keywords and phrases Query Complexity, Communication Complexity, Subcube Partition Complexity, Partition Bound

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.4

1 Introduction

Deterministic communication complexity

In the standard model of communication complexity, we wish to compute a function $F : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$, where the inputs $x \in \mathcal{X}$ and $y \in \mathcal{Y}$ are given to two different players, while minimizing the communication between the players. We use $D^{\text{cc}}(F)$ to denote the deterministic communication complexity of F, the number of bits communicated in the worst case by the best deterministic protocol for the function F.

The partition number of F, denoted $\chi(F)$, is the least number of monochromatic rectangles in a partition or disjoint cover of $\mathcal{X} \times \mathcal{Y}$ (where a monochromatic rectangle is a set $A \times B$, with $A \subseteq \mathcal{X}$ and $B \subseteq \mathcal{Y}$, such that F takes the same value on all elements of $A \times B$). Yao [30] observed that any C-bit communication protocol for F partitions the set of all inputs $\mathcal{X} \times \mathcal{Y}$ into at most 2^C monochromatic rectangles, which gives us $\log \chi(F) \leq D^{\text{cc}}(F)$. This

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 4; pp. 4:1–4:14


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

Table 1 Known separations between deterministic communication complexity, $D^{cc}(F)$, and partition number, $\chi(F)$.

Separation	Reference
$D^{cc}(F) \geq 2\log\chi(F)$	[18]
$D^{cc}(F) = \widetilde{\Omega}\left(\log^{1.5}\chi(F)\right)$	[10]
$D^{cc}(F) \geq \left(\log\chi(F)\right)^{2-o(1)}$	Theorem 1.1
$D^{cc}(F) = O\left(\log\chi(F)^2\right)$ for all $F : \mathcal{X}\times\mathcal{Y}\to\{0,1\}$	

turns out to be a powerful lower bound, and in fact almost all lower bound techniques for deterministic communication complexity, including the partition bound, discrepancy, fooling sets, (nonnegative) rank, and various norm-based methods [13, 14, 22], actually lower bound $\log\chi(F)$.

In addition to being a fruitful lower bound technique, $\log\chi(F)$ also yields an upper bound on $D^{cc}(F)$. Aho, Ullman, and Yannakakis [4] showed that for all $F : \mathcal{X}\times\mathcal{Y}\to\{0,1\}$, we have

$$D^{cc}(F) = O(\log^2\chi(F)). \tag{1}$$

It has been a long-standing open problem to determine whether this upper bound can be improved (see, e.g., [19, Open Problem 2.10]). We show that the upper bound in (1) is essentially optimal.

▶ **Theorem 1.1.** *There exists a function $F : \mathcal{X}\times\mathcal{Y}\to\{0,1\}$ with $D^{cc}(F) \geq \left(\log\chi(F)\right)^{2-o(1)}$.*

Until recently, the best known separation between the two measures was only by a factor of 2 [18]. Recently, Göös, Pitassi, and Watson [10] showed that there exists a function F with $D^{cc}(F) = \widetilde{\Omega}(\log^{1.5}\chi(F))$, where the notation $\widetilde{\Omega}(m)$ hides $\mathrm{poly}(\log m)$ factors. Table 1 summarizes known separations between D^{cc} and $\log\chi$.

Deterministic query complexity

In the model of query complexity, we wish to compute a function $f : \{0,1\}^n\to\{0,1\}$ on an input $x\in\{0,1\}^n$ given query access to the bits of the input, i.e., we can only access the input via a black box that accepts an index $i\in[n]$ (where $[n] := \{1,2,\ldots,n\}$) and responds with $x_i\in\{0,1\}$. The goal is to compute $f(x)$ while minimizing the number of queries made to the black box. Let $D(f)$ to denote the deterministic query complexity of f, the number of queries made by the best deterministic algorithm that computes f correctly on all inputs.

As in communication complexity, most lower bounds for deterministic query complexity are based on the simple observation that any d-query algorithm computing f partitions the domain $\{0,1\}^n$ into at most 2^d monochromatic subcubes where each subcube fixes at most d variables. A subcube is a restriction of the hypercube where some variables have been fixed, and it is monochromatic if f takes the same value on all inputs in the subcube. This motivates defining the subcube partition complexity of f as a smallest d such that the domain $\{0,1\}^n$ can be partitioned into at most 2^d monochromatic subcubes that each fix at most d variables. Subcube partition complexity can also be viewed as an unambiguous version of certificate complexity as explained in Section 3, and hence we denote this measure $\mathrm{UC}(f)$.

Table 2 Known separations between deterministic query complexity and subcube partition complexity.

Separation	Reference
$D(f) = \Omega(\mathrm{UC}(f)^{1.261})$	[27]
$D(f) = \widetilde{\Omega}(\mathrm{UC}(f)^{1.5})$	[10]
$D(f) \geq \mathrm{UC}(f)^{2-o(1)}$	Theorem 1.2
$D(f) = O(\mathrm{UC}(f)^2)$ for all $f : \{0,1\}^n \to \{0,1\}$	

Table 3 Known separations between randomized query complexity and subcube partition complexity.

Separation	Reference
$R(f) = \Omega(\mathrm{UC}(f)^{1.058})$	[17]
$R(f) = \widetilde{\Omega}(\mathrm{UC}(f)^{1.5})$	[9]
$R(f) \geq \mathrm{UC}(f)^{2-o(1)}$	Theorem 1.3
$R(f) = O(\mathrm{UC}(f)^2)$ for all $f : \{0,1\}^n \to \{0,1\}$	

Due to the observation above, we have $\mathrm{UC}(f) \leq D(f)$. It turns out that this lower bound is also relatively tight: for all $f : \{0,1\}^n \to \{0,1\}$ we have

$$D(f) = O(\mathrm{UC}(f)^2). \tag{2}$$

We show that this upper bound is essentially optimal.

▶ **Theorem 1.2.** *There exists a total function f with $D(f) \geq \mathrm{UC}(f)^{2-o(1)}$.*

The first separation between these two measures was a power 1.261 separation by Savický, which was recently improved by Göös, Pitassi, and Watson [10] to power 1.5. Table 2 summarizes known separations between these measures.

Randomized query complexity

We can extend the query model to allow randomized algorithms in the natural way. We define the bounded-error randomized query complexity of a function f, $R(f)$, to be the minimum number of queries needed in the worst case by a randomized algorithm that outputs $f(x)$ on input x with probability at least $2/3$.

As before, almost all lower bound techniques for randomized query complexity are upper bounded by $\mathrm{UC}(f)$, as shown in [17]. This includes the partition bounds [13, 14], approximate polynomial degree [25], approximate nonnegative junta degree (also known as nonnegative literal degree or conical junta degree) [16], block sensitivity [24], randomized certificate complexity or fractional block sensitivity [1, 7, 29], and the classical analogue of the quantum adversary bound [20, 28, 2].

Since we obviously have $R(f) \leq D(f)$, using (2) we know that $R(f) = O(\mathrm{UC}(f)^2)$. We show that this upper bound is also essentially optimal.

▶ **Theorem 1.3.** *There exists a total function f with $R(f) \geq \mathrm{UC}(f)^{2-o(1)}$.*

The first asymptotic separation between these measures was a power 1.058 separation by Racicot-Desloges, Santha, and Kothari [17], which was later improved by Göös, Jayram, Pitassi, and Watson [9] to a power 1.5 separation. Table 3 summarizes the known separations between these measures.

Quantum query complexity

The query model can also be naturally extended to quantum algorithms. We denote by $Q(f)$ the bounded-error quantum query complexity of f, the minimum number of queries made in the worst case by a quantum algorithm that outputs $f(x)$ on input x with probability at least $2/3$. (See [6] for a formal definition.)

As before, since $Q(f) \leq D(f)$, using (2) we know that $Q(f) = O(\mathrm{UC}(f)^2)$. However, prior to our work no function was known for which $Q(f) \gg \mathrm{UC}(f)$ was known. Furthermore, the functions previously used to show separations between $D(f)$ or $R(f)$ and $\mathrm{UC}(f)$ do not separate $Q(f)$ from $\mathrm{UC}(f)$. Indeed, even the functions constructed to prove Theorem 1.2 and Theorem 1.3 do not separate $Q(f)$ from $\mathrm{UC}(f)$. Despite this, we give the first superlinear separation between $Q(f)$ and $\mathrm{UC}(f)$.

▶ **Theorem 1.4.** *There exists a total function f with $Q(f) \geq \mathrm{UC}(f)^{1.5-o(1)}$.*

We are also able to show an improved separation between quantum query complexity and one-sided subcube partition complexity, denoted by $\mathrm{UC}_1(f)$, which is similar to subcube partition complexity except that we only need to partition the 1-inputs using monochromatic subcubes.

For this measure, the quadratic upper bound $D(f) = O(\mathrm{UC}_1(f)^2)$ still holds [8, Proposition 5], and hence $Q(f) = O(\mathrm{UC}_1(f)^2)$. We show this upper bound is optimal up to log factors, qualitatively improving upon [10] and [9] who proved the same result for deterministic and randomized query complexity respectively.

▶ **Theorem 1.5.** *There exists a total function f with $Q(f) = \widetilde{\Omega}(\mathrm{UC}_1(f)^2)$.*

2 High-level overview

We now provide a high-level overview of the separations shown.

Deterministic communication complexity

We prove Theorem 1.1 by showing the analogous separation in query complexity (Theorem 1.2) and "lifting" the result to communication complexity, which is also the strategy used in [10]. Essentially, the deterministic simulation theorem of [10] provides a black-box way of converting a query separation between $D(f)$ and $\mathrm{UC}(f)$ to a separation between $D^{\mathrm{cc}}(F)$ and $\log \chi(F)$. The theorem weakens the separation by log of the input size of f, but with a suitable choice of parameters this is negligible compared to the $o(1)$ term in the separation.

Deterministic query complexity

To prove Theorem 1.2, we use the recently introduced cheat sheet framework [3] and the commonly used technique of function composition. Before describing the construction, we need to define some notation. For any functions $f : \{0,1\}^n \to \{0,1\}$ and $g : \{0,1\}^m \to \{0,1\}$, we define the composed function $f \circ g$ to be the function on mn bits whose output on $y = (y_{11}, \ldots, y_{1m}, \ldots, y_{n1}, \ldots, y_{nm})$ is $f(g(y_{11}, \ldots, y_{1m}), \ldots, g(y_{n1}, \ldots, y_{nm}))$. Let $\textsc{And}_n$ and $\textsc{Or}_n$ denote the And and Or function on n bits respectively. For any function f, we use f_{CS} to denote the "cheat sheet version" of f, a new total Boolean function constructed from f. (We review the cheat sheet framework in Section 4.)

An interesting feature of the cheat sheet framework is that $\mathrm{UC}_1(f_{\mathrm{CS}})$ can be substantially smaller than $\mathrm{UC}_1(f)$ because one can construct a partition for inputs with $f_{\mathrm{CS}} = 1$ without using a partition for inputs with $f = 1$. This property is crucial for our construction but is not sufficient by itself because the complexity of the best partition for inputs with $f_{\mathrm{CS}} = 0$ is of a similar order as $\mathrm{UC}(f)$. To deal with this, we combine the cheat sheet construction with several other steps which rebalance the complexity of partitions for $f = 1$ and $f = 0$.

We construct our function in stages starting with the function $f_0 = \textsc{And}_n$ that achieves no separation between $D(f)$ and $\mathrm{UC}(f)$. We then compose the function with $\textsc{Or}_n$, construct

the cheat sheet version, and then compose with AND_n, to obtain the function $f_1 = \text{AND}_n \circ (\text{OR}_n \circ \text{AND}_n)_{\text{CS}}$, which achieves a power 3/2 separation between $D(f)$ and $\text{UC}(f)$. Repeating this construction once more yields $f_2 = \text{AND}_n \circ (\text{OR}_n \circ \text{AND}_n \circ (\text{OR}_n \circ \text{AND}_n)_{\text{CS}})_{\text{CS}}$, which achieves a power 5/3 separation, and so on. The function f_k achieves a $(2k+1)/(k+1)$ separation, which yields a $2 - o(1)$ separation if we choose k to be a slow growing function of n.

Randomized query complexity

The function constructed above also yields the separation in Theorem 1.3 with slightly worse parameters. The analysis of the constructed function is similar since deterministic and randomized query complexities behave similarly with respect to the cheat sheet technique and with respect to composition with the AND and OR functions.

Quantum query complexity

Lastly, we establish the quantum separations using two functions introduced by Aaronson, Ben-David and Kothari [3]: the BLOCK-k-SUM-OF-k-SUMS function, which we denote BKK, and the BLOCK-k-SUM function, which we denote BK-SUM. The function BKK_{CS} yields the separation in Theorem 1.5. The separation in Theorem 1.4 requires a function constructed in stages again. The first function is $f_1 = \text{AND} \circ \text{BKK}_{\text{CS}}$, which achieves a power 5/4 separation, the next is $f_2 = \text{AND}_n \circ (\text{BK-SUM}_n \circ f_1)$, which achieves a power 4/3 separation and so on. The function f_k achieves a power $(3k+2)/(2k+2)$ separation.

3 Preliminaries

Communication complexity

The only communication complexity measures we need are $D^{\text{cc}}(F)$ and $\chi(F)$, which were defined in Section 1. The interested reader is referred to [19, 15] for more formal definitions of these measures.

Query complexity

For more formal definitions of measures introduced in Section 1, the reader is referred to the survey by Buhrman and de Wolf [6]. The only measure not covered in the survey is subcube partition complexity, which is explained in detail in [17].

Subcube partition complexity can also be viewed as unambiguous certificate complexity and we use this perspective in this paper. To explain this, let us begin with certificate complexity.

A certificate for an input $x \in \{0,1\}^n$ is a subset $S \subseteq [n]$ of indices and claimed values for these bits, such that x is consistent with the certificate and any input y consistent with the certificate satisfies $f(x) = f(y)$. In other words, a certificate for x is a partial assignment of bits consistent with x such that any other string consistent with this partial assignment has the same function value as x. For $b \in \{0,1\}$, the b-certificate complexity of f, denoted $C_b(f)$, is the size of the smallest certificate for x maximized over all inputs with $f(x) = b$. The certificate complexity of f, $C(f)$, is defined as $C(f) := \max\{C_0(f), C_1(f)\}$. Alternately, $C_1(f)$ is the smallest w such that f can be written as a width-w DNF, i.e., a DNF in which each term contains at most w variables. Similarly, $C_0(f)$ corresponds to CNF width.

Unambiguous certificate complexity is defined similarly, except we require the set of certificates to be unambiguous, i.e., at most one certificate from the set of all certificates should work for a given input. In other words, the unambiguous 1-certificate complexity of f is the minimum w such that f can be written as a width-w DNF in which at most one term evaluates to 1 on any input. Similar to certificate complexity, we denote unambiguous b-certificate complexity by $\mathrm{UC}_b(f)$ and define $\mathrm{UC}(f) := \max\{\mathrm{UC}_0(f), \mathrm{UC}_1(f)\}$. Clearly, since unambiguous certificates are more restricted than certificates, we have for $b \in \{0,1\}$, $C_b(f) \leq \mathrm{UC}_b(f)$ and $C(f) \leq \mathrm{UC}(f)$.

For example, consider the OR_n function on n bits defined as $\bigvee_{i\in[n]} x_i$. Clearly $C_0(\mathrm{OR}_n) = n$ since we must examine all n bits to be sure that all $x_i = 0$. On the other hand, $C_1(\mathrm{OR}_n) = 1$ since the location of any 1 in the input is a certificate. Obviously $\mathrm{UC}_0(\mathrm{OR}_n)$ remains n. However, a single 1 in the input is not an unambiguous 1-certificate since inputs with multiple 1s would have multiple valid certificates. In other words, although $\bigvee_{i\in[n]} x_i$ is a valid DNF representation of OR_n, it is not unambiguous since several terms can simultaneously be 1. So consider the following DNF:

$$\mathrm{OR}_n(x) = x_1 \vee \overline{x_1}x_2 \vee \overline{x_1x_2}x_3 \vee \cdots \vee \overline{x_1x_2\cdots x_{n-1}}x_n\,. \tag{3}$$

This DNF is unambiguous since any term evaluating to 1 prevents other terms from evaluating to 1. Thus we have $\mathrm{UC}_1(\mathrm{OR}_n) \leq n$. Although this result in trivial because $\mathrm{UC}_1(f) \leq n$ for any n-bit function f, this DNF representation of OR_n will be useful to us later later because it has the property that every unambiguous certificate has only one unnegated index x_i.

Composition theorems

Composition theorems relate the complexity of composed functions with the complexities of the individual functions. For example, for all Boolean functions f and g, $D(f\circ g) = D(f)D(g)$ [29, 23]. In our construction we will repeatedly compose functions with AND_n and OR_n, and hence we need to understand the complexities of the resulting functions.

▶ **Lemma 3.1** (AND/OR composition). *For any total Boolean function f, the following bounds hold:*

- $D(\mathrm{AND}_n \circ f) = nD(f)$
- $R(\mathrm{AND}_n \circ f) = \Omega(nR(f))$
- $Q(\mathrm{AND}_n \circ f) = \Omega(\sqrt{n}Q(f))$
- $C_0(\mathrm{AND}_n \circ f) \leq C_0(f)$
- $C_1(\mathrm{AND}_n \circ f) \leq nC_1(f)$
- $\mathrm{UC}_0(\mathrm{AND}_n \circ f) \leq \mathrm{UC}_0(f) + (n-1)\mathrm{UC}_1(f)$
- $\mathrm{UC}_1(\mathrm{AND}_n \circ f) \leq n\mathrm{UC}_1(f)$
- $D(\mathrm{OR}_n \circ f) = nD(f)$
- $R(\mathrm{OR}_n \circ f) = \Omega(nR(f))$
- $Q(\mathrm{OR}_n \circ f) = \Omega(\sqrt{n}Q(f))$
- $C_0(\mathrm{OR}_n \circ f) \leq nC_0(f)$
- $C_1(\mathrm{OR}_n \circ f) \leq C_1(f)$
- $\mathrm{UC}_0(\mathrm{OR}_n \circ f) \leq n\mathrm{UC}_0(f)$
- $\mathrm{UC}_1(\mathrm{OR}_n \circ f) \leq (n-1)\mathrm{UC}_0(f) + \mathrm{UC}_1(f)$

Proof. We prove the claims for the function $\mathrm{AND}_n \circ f$. Similar reasoning proves the analogous claims for the function $\mathrm{OR}_n \circ f$.

The first property follows from the fact that $D(f\circ g) = D(f)D(g)$ for any Boolean functions f and g [29, 23]. $R(\mathrm{AND}_n \circ f) = \Omega(nR(f))$ was recently proved by [9]. $Q(\mathrm{AND}_n \circ f) = \Omega(\sqrt{n}Q(f))$ because $Q(f \circ g) = \Theta(Q(f)Q(g))$ for any Boolean functions f and g [12, 26, 21] and we know that $Q(\mathrm{AND}_n) = Q(\mathrm{OR}_n) = \Theta(\sqrt{n})$ [11, 5].

We have $C_0(\mathrm{AND}_n \circ f) \leq C_0(f)$ since a 0-certificate for AND_n is a 0-input to it, which corresponds to an instance of f that evaluates to 0. On the other hand, by certifying that all n instances of f evaluate to 1, we can certify $\mathrm{AND}_n \circ f$ evaluates to 1, and hence $C_1(\mathrm{AND}_n \circ f) \leq nC_1(f)$.

We can unambiguously certify that $\text{AND}_n \circ f$ evaluates to 0 by unambiguously certifying the value of the first (from the left) 1-input to the AND_n gate and unambiguously certifying that all previous inputs are 0. This is the same idea used to construct the unambiguous DNF for OR_n in (3). This construction gives $\text{UC}_0(\text{AND}_n \circ f) \leq \text{UC}_0(f) + (n-1)\text{UC}_1(f)$. We can unambiguously certify that $\text{AND}_n \circ f$ evaluates to 1 by providing unambiguous 1-certificates for all n instances of f. This gives $\text{UC}_1(\text{AND}_n \circ f) \leq n\text{UC}_1(f)$. ◀

4 Cheat sheet framework

We now overview the recently introduced cheat sheet framework [3]. The framework as presented in [3] is more general and can fulfill different objectives such as making partial functions total. We present a restricted version of the framework that only works for total functions. We use the framework because it makes 1-certificates unambiguous in a natural way.

▶ **Definition 4.1** (Cheat sheet version of a total function)**.** Let $f : \{0,1\}^N \to \{0,1\}$ be a function, $c = 10 \log N$ and $m = 10C(f) \log^2 N$. Then the *cheat sheet version of* f, denoted f_{CS}, is a total function

$$f_{\text{CS}} : (\{0,1\}^N)^c \times (\{0,1\}^m)^{2^c} \to \{0,1\}.$$

Let the input be written as $(x^1, x^2, \ldots, x^c, Y_1, Y_2, \ldots, Y_{2^c})$, where for all $i \in [N]$, $x^i \in \{0,1\}^N$ and for all $j \in [2^c]$, $Y_j \in \{0,1\}^m$. Let $\ell_i = f(x^i)$ and $\ell \in [2^c]$ be the positive integer corresponding to the binary string $\ell_1 \ell_2 \ldots \ell_c$. Then we define the value of $f_{\text{CS}}(x^1, x^2, \ldots, x^c, Y_1, Y_2, \ldots, Y_{2^c})$ to be 1 if and only if Y_ℓ contains certificates for $f(x^i) = \ell_i$ for all $i \in [c]$.

Informally, the cheat sheet construction takes any total function f and converts it into a new total function f_{CS} in the following way. An input to the new function f_{CS} first contains $c = 10 \log N$ inputs to f and then a vast array of size 2^c of cells of size m bits. The outputs of these c inputs to f is a bit string $\ell_1 \ell_2 \ldots \ell_c$ of length c that represents an integer $\ell \in [2^c]$ in the natural way. We treat this integer ℓ as an address into this array of size 2^c and say that these c inputs to f point to the ℓ^{th} cell of the array. At the ℓ^{th} cell of the array we require certificates certifying that this was indeed the cell pointed to by the c inputs to f. In other words, we require certificates certifying that $f(x^i)$, the output of f acting on the i^{th} input, is indeed equal to ℓ_i for all $i \in [c]$. Since a certificate for a single f consists of $C(f)$ pointers to the input, a certificate is of size $C(f) \log N$ bits, and hence c certificates are of size $m = C(f)c \log N = 10C(f) \log^2 N$. The function f_{CS} is defined to be 1 if and only if the input satisfies this property, i.e., if the cell pointed to by the c instances does indeed contain certificates certifying it is the correct cell.

This construction preserves the complexity of f with respect to some measures. For example, $D(f_{\text{CS}})$ equals $D(f)$ up to log factors. The upper bound uses the natural algorithm for f_{CS}: the deterministic algorithm first computes the c copies of f on inputs x^1 to x^c and finds the cell pointed to by these c inputs. Then it checks if the certificates in this cell certify that this is the right cell. This requires $cD(f)$ queries to compute the c copies, m queries to read the contents of the cell and $cC(f)$ queries to check if the certificates are all correct. Overall this uses $O(cD(f))$ queries. We also have $D(f_{\text{CS}}) = \Omega(D(f))$, because intuitively if an algorithm cannot compute f it has no hope of finding the cheat sheet since that would require solving c copies of f or searching in an array of size n^{10}. Similarly, many measures behave as expected under cheat sheets, and we show this below.

▶ **Lemma 4.2** (Complexity of cheat sheet functions). *For any total function $f : \{0,1\}^N \to \{0,1\}$, if $f_{\mathrm{CS}} : \{0,1\}^{N'} \to \{0,1\}$ denotes the cheat sheet version of f as defined in Definition 4.1, then we have the following upper and lower bounds:*

- $D(f_{\mathrm{CS}}) = \Omega(D(f))$
- $R(f_{\mathrm{CS}}) = \Omega(R(f)/\log^2 N)$
- $Q(f_{\mathrm{CS}}) = \Omega(Q(f))$
- $C_0(f_{\mathrm{CS}}) = O(C(f)\log^2 N)$
- $C_1(f_{\mathrm{CS}}) = O(C(f)\log^2 N)$
- $\mathrm{UC}_0(f_{\mathrm{CS}}) = O(\mathrm{UC}(f)\log^2 N)$
- $\mathrm{UC}_1(f_{\mathrm{CS}}) = O(C(f)\log^2 N)$
- $N' = O(N^{12})$

Proof. We have $D(f_{\mathrm{CS}}) = \Omega(D(f))$ [3, Lemma 21], $R(f_{\mathrm{CS}}) = \Omega(R(f)/\log^2 N)$ [3, Lemma 6], and $Q(f_{\mathrm{CS}}) = \Omega(Q(f))$ [3, Lemma 12].

We have $C_0(f_{\mathrm{CS}}) = O(C(f)\log^2 N)$ because a valid 0-certificate for f_{CS} can first certify the c outputs to f, which requires $O(cC(f))$ queries. This points to a cell ℓ. The certificate can then contain the contents of cell ℓ of size $O(C(f)\log^2 N)$ and the locations pointed to (and the bits contained at these locations) by the certificates in cell ℓ. After querying this cell and all the locations pointed to by the certificates in this cell, it can be determined with no further queries if this cell is incorrectly filled. We have $C_1(f_{\mathrm{CS}}) = O(C(f)\log^2 N)$ since the location of the correct cell and the pointers within that cell along with the bits they point to forms a 1-certificate.

We have $\mathrm{UC}_0(f_{\mathrm{CS}}) = O(\mathrm{UC}(f)\log^2 N)$ using the same argument as for certificate complexity. We first certify the c outputs to f unambiguously using unambiguous certificates of size $\mathrm{UC}(f)$. This points to a cell ℓ. The certificate also contains the contents of cell ℓ and the locations pointed to (and the bits at these locations) by the certificates in cell ℓ. This certificate is unambiguous because this certificate evaluating to true prevents any other certificate from evaluating to true. To see this, note that if another certificate tries to certify a different value of ℓ then this will be an invalid certificate. If the certificate claims the same value of ℓ, then it must use the same certificates for the c instances of f because we used unambiguous certificates and hence there is only one valid certificate for each $f(x^i) = \ell_i$. Now if the other certificate has the same value of ℓ but different claimed values for the contents of the ℓ^{th} cell or the locations pointed to by the cell, this will be inconsistent with the actual input since our original certificate was consistent with the input.

We have $\mathrm{UC}_1(f_{\mathrm{CS}}) = O(C(f)\log^2 N)$. For this case an unambiguous certificate will contain only the contents of cell ℓ and the locations pointed to by the certificates in cell ℓ along with the bits contained at these locations. This is identical to the 1-certificate we constructed above. Since this is clearly a valid certificate, we only need to show it is unambiguous, i.e., that if this certificate evaluates to true, all other certificates must fail. If another certificate has a different value of ℓ, then its contents will not be able to certify that the output of the c functions equals ℓ and the certificate will be rejected. On the other hand, if the other certificate has the same value of ℓ but different claimed values for the contents of the cell or the locations pointed to by the cell, this will be inconsistent with the input since our original certificate was consistent with the input.

Lastly, we need to upper bound the input size of f_{CS}. From Definition 4.1 we know the input size is $cN + m2^c = 10N\log N + 10N^{10}C(f)\log^2 N = O(10N^{11}\log^2 N) = O(N^{12})$. ◀

5 Randomized query complexity vs. subcube partitions and deterministic communication vs. partition number

Randomized query complexity vs. subcube partitions

We now establish the following theorem which implies Theorem 1.3, which in turn implies Theorem 1.2.

▶ **Theorem 5.1.** *For every $k \geq 0$, there exists a total Boolean function $f_k : \{0,1\}^{N_k} \to \{0,1\}$, such that $R(f_k) = \widetilde{\Omega}(n^{2k+1})$ and $\mathrm{UC}(f_k) = \widetilde{O}(n^{k+1})$. Hence there is a function f with $R(f) \geq \mathrm{UC}(f)^{2-o(1)}$.*

Proof. Let $f_0 = \textsc{And}_n$ and f_k be defined inductively as $f_k := \textsc{And}_n \circ (\textsc{Or}_n \circ f_{k-1})_{\mathrm{CS}}$, i.e., f_k is the function obtain by composing $\textsc{And}_n$ with the cheat sheet version of $\textsc{Or}_n$ composed with f_{k-1}.

We prove the claim by induction on k. The induction hypothesis and the base case, $f_0 = \textsc{And}_n$, are presented below, where N_k is the input size of the function f_k.

Induction hypothesis (f_k)
- $N_k = O(n^{25^k})$
- $D(f_k) = \widetilde{\Omega}(n^{2k+1})$
- $R(f_k) = \widetilde{\Omega}(n^{2k+1})$
- $C_0(f_k) = \widetilde{O}(n^k)$
- $C_1(f_k) = \widetilde{O}(n^{k+1})$
- $\mathrm{UC}_0(f_k) = \widetilde{O}(n^{k+1})$
- $\mathrm{UC}_1(f_k) = \widetilde{O}(n^{k+1})$

Base case ($f_0 = \textsc{And}_n$)
- $N_0 = n$
- $D(f_0) = n$
- $R(f_0) = \Omega(n)$
- $C_0(f_0) \leq 1$
- $C_1(f_0) \leq n$
- $\mathrm{UC}_0(f_0) \leq n$
- $\mathrm{UC}_1(f_0) \leq n$

The complexities of $f_0 = \textsc{And}_n$ are straightforward to show and also follow from the general composition lemma (Lemma 3.1) by letting f be the one-bit identity function. Clearly the base case is consistent with the induction hypothesis.

We now show that the induction hypothesis for f_k implies the same for f_{k+1}. First we upper bound the input size of $f_{k+1} = \textsc{And}_n \circ (\textsc{Or}_n \circ f_k)_{\mathrm{CS}}$. Since the input size of f_k is $O(n^{25^k})$, the input size of $\textsc{Or}_n \circ f_k$ is $O(n^{25^k+1})$ and the input size of $(\textsc{Or}_n \circ f_k)_{\mathrm{CS}}$ is $O(n^{12(25^k+1)})$ (from Lemma 4.2). Hence the input size of f_{k+1} is $O(n^{12(25^k+1)+1}) = O(n^{12(25^k)+13}) = O(n^{25^{k+1}})$.

The deterministic query complexity of f_{k+1} can be lower bounded as follows:

$$D(f_{k+1}) = D(\textsc{And}_n \circ (\textsc{Or}_n \circ f_k)_{\mathrm{CS}}) = nD((\textsc{Or}_n \circ f_k)_{\mathrm{CS}}) = \Omega(nD(\textsc{Or}_n \circ f_k)) = \widetilde{\Omega}(n^{2k+3}),$$

where we used Lemma 3.1 and Lemma 4.2 to compute the relevant measures. The same calculation also works for $R(f_{k+1})$ up to log factors since $R(f)$ and $D(f)$ behave similarly in the aforementioned lemmas up to log factors. Similarly using Lemma 3.1 and Lemma 4.2 we have

$$C_0(f_{k+1}) = C_0(\textsc{And}_n \circ (\textsc{Or}_n \circ f_k)_{\mathrm{CS}}) \leq C_0((\textsc{Or}_n \circ f_k)_{\mathrm{CS}}) = \widetilde{O}(C(\textsc{Or}_n \circ f_k)) = \widetilde{O}(n^{k+1})$$

and

$$C_1(f_{k+1}) = C_1(\textsc{And}_n \circ (\textsc{Or}_n \circ f_k)_{\mathrm{CS}}) \leq nC_1((\textsc{Or}_n \circ f_k)_{\mathrm{CS}}) = \widetilde{O}(nC(\textsc{Or}_n \circ f_k)) = \widetilde{O}(n^{k+2}).$$

In these bounds we do not differentiate between $\log N_k$ and $\log n$ because they are asymptotically equal, since $\log N_k = 25^k \log n = O(\log n)$. Finally, using Lemma 3.1 and

Lemma 4.2 again we have

$$\begin{aligned}
\mathrm{UC}_0(f_{k+1}) &= \mathrm{UC}_0(\textsc{And}_n \circ (\textsc{Or}_n \circ f_k)_{\mathrm{CS}}) \\
&\leq \max\{\mathrm{UC}_0((\textsc{Or}_n \circ f_k)_{\mathrm{CS}}), n\mathrm{UC}_1((\textsc{Or}_n \circ f_k)_{\mathrm{CS}})\} \\
&= \widetilde{O}\left(\max\{\mathrm{UC}(\textsc{Or}_n \circ f_k), nC(\textsc{Or}_n \circ f_k)\}\right) = \widetilde{O}(n^{k+2}) \qquad \text{and} \\
\mathrm{UC}_1(f_{k+1}) &= \mathrm{UC}_1(\textsc{And}_n \circ (\textsc{Or}_n \circ f_k)_{\mathrm{CS}}) \leq n\mathrm{UC}_1((\textsc{Or}_n \circ f_k)_{\mathrm{CS}}) \\
&= \widetilde{O}(nC(\textsc{Or}_n \circ f_k)) = \widetilde{O}(\max\{nC_0(f_k), C_1(f_k)\} = \widetilde{O}(n^{k+2}).
\end{aligned}$$

This completes the induction and establishes the first part of the theorem.

For the second part, since $R(f_k) = \widetilde{\Omega}(n^{2k+1})$ and $\mathrm{UC}(f_k) = \widetilde{O}(n^{k+1})$, we have $R(f_k) = \widetilde{\Omega}(\mathrm{UC}(f_k)^{2-\frac{1}{k+1}})$. Since we treated k as a constant, our notation hides constant and $\log n$ factors that depend only on k, i.e., we only get $R(f_k) \geq \left(\mathrm{UC}(f_k)^{2-\frac{1}{k+1}}\right) \Big/ \left(h_1(k)\log^{h_2(k)} n\right)$ for some functions $h_1(k)$ and $h_2(k)$. But we can always choose k to be a slow growing function of n so that these terms are negligible. This yields the desired separation $R(f) \geq \mathrm{UC}(f)^{2-o(1)}$. ◀

Clearly Theorem 1.2 and Theorem 1.3 follow from this, which we restate for convenience.

▶ **Theorem 1.2.** *There exists a total function f with $D(f) \geq \mathrm{UC}(f)^{2-o(1)}$.*

▶ **Theorem 1.3.** *There exists a total function f with $R(f) \geq \mathrm{UC}(f)^{2-o(1)}$.*

Deterministic communication vs. partition number

We now show Theorem 1.1 by lifting the previous separation to communication complexity.

From Theorem 1.3, we have a function $f : \{0,1\}^N \to \{0,1\}$ such that $R(f) \geq \mathrm{UC}(f)^{2-o(1)}$, which implies $D(f) \geq \mathrm{UC}(f)^{2-o(1)}$. Göös, Pitassi, and Watson [10] show that for any function f, there is a corresponding communication problem F such that

$$D^{\mathrm{cc}}(F) = \Omega(D(f)\log N) = \Omega(D(f)).$$

On the other hand, as explained in [10], we also have

$$\log \chi(F) = O(\mathrm{UC}(f)\log N) = \widetilde{O}(\mathrm{UC}(f)),$$

where we used the fact that our function has $N = n^{25^k}$, where k is a slow growing function of n, and hence $\log N = 25^k \log n = O(\log^2 n) = O(\log^2 \mathrm{UC}(f))$.

Since the conversion to communication complexity only weakens the result by log factors, the separation $D(f) \geq \mathrm{UC}(f)^{2-o(1)}$ immediately yields

$$D^{\mathrm{cc}}(F) \geq \left(\log \chi(F)\right)^{2-o(1)},$$

which establishes Theorem 1.1:

▶ **Theorem 1.1.** *There exists a function $F : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ with $D^{\mathrm{cc}}(F) \geq \left(\log \chi(F)\right)^{2-o(1)}$.*

6 Quantum query complexity vs. subcube partitions

In this section we establish Theorem 1.4 and Theorem 1.5. To show this we require a function Bkk from [3, Theorem 10] with the following properties.

▶ **Lemma 6.1.** *There exists a total function* $\textsc{Bkk} : \{0,1\}^{n^2} \to \{0,1\}$ *such that* $C(\textsc{Bkk}) = \widetilde{O}(n)$ *and* $Q(\textsc{Bkk}) = \widetilde{\Omega}(n^2)$.

We are now ready to prove Theorem 1.5, restated for convenience:

▶ **Theorem 1.5.** *There exists a total function* f *with* $Q(f) = \widetilde{\Omega}(\mathrm{UC}_1(f)^2)$.

Proof. Let $f = \textsc{Bkk}$. Then using Lemma 4.2 we know that $Q(f_{\mathrm{CS}}) = \Omega(Q(f)) = \widetilde{\Omega}(n^2)$ and $\mathrm{UC}_1(f_{\mathrm{CS}}) = O(C(f)\log^2 n) = \widetilde{O}(n)$. ◀

To show Theorem 1.4, we need another function BK-SUM, which is a variant of the K-SUM problem. It has the interesting property that any certificates for it consists essentially of input bits set to 0 and very few input bits set to 1. As shown in the proof of [3, Theorem 10], we have the following. (More precisely, our version of BK-SUM swaps the roles of zeros and ones compared to the function of [3], but this does not affect its quantum query complexity.)

▶ **Lemma 6.2.** *There exists a total function* $\textsc{bk-sum} : \{0,1\}^n \to \{0,1\}$ *with* $Q(\textsc{bk-sum}) = \widetilde{\Omega}(n)$ *such that for any function* f*, we have* $C(\textsc{bk-sum} \circ f) = O(nC_0(f) + C_1(f)\log^3 n)$.

In our construction, we repeatedly compose BK-SUM with other functions and hence we need to understand the behavior of BK-SUM under composition, analogous to Lemma 3.1 for AND and OR.

▶ **Lemma 6.3** (BK-SUM composition)**.** *For any function* f*, the following bounds hold:*

- $Q(\textsc{bk-sum}_n \circ f) = \widetilde{\Omega}(nQ(f))$
- $C(\textsc{bk-sum}_n \circ f) = O(nC_0(f) + C_1(f)\log^3 n)$.
- $\mathrm{UC}(\textsc{bk-sum}_n \circ f) \leq n\mathrm{UC}(f)$

Proof. The first lower bound follows because $Q(f \circ g) = \Theta(Q(f)Q(g))$ for any Boolean functions f and g [12, 26, 21] and we have $Q(\textsc{bk-sum}) = \widetilde{\Omega}(n)$ from Lemma 6.2. The second relation follows from Lemma 6.2. Lastly, $\mathrm{UC}(\textsc{bk-sum}_n \circ f) \leq n\mathrm{UC}(f)$ because this holds for any n-bit function. Any function $h \circ f$ can be unambiguously certified by showing all the outputs to f and providing unambiguous certificates for each output. ◀

We are now ready to establish the following theorem, which implies Theorem 1.4. This proof mimics the proof structure of Theorem 5.1 and reuses several ideas.

▶ **Theorem 6.4.** *For every* $k \geq 0$*, there exists a total Boolean function* $f_k : \{0,1\}^{N_k} \to \{0,1\}$*, such that* $Q(f_k) = \widetilde{\Omega}(n^{1.5k+1})$ *and* $\mathrm{UC}(f_k) = \widetilde{O}(n^{k+1})$*. Hence there is a function* f *with* $Q(f) \geq \mathrm{UC}(f)^{1.5-o(1)}$.

Proof. Let $f_1 = \textsc{And}_n \circ \textsc{Bkk}_{\mathrm{CS}}$, where BKK is the function on n^2 bits in Lemma 6.1. Let f_k be defined inductively as $f_k := \textsc{And}_n \circ (\textsc{bk-sum}_n \circ f_{k-1})_{\mathrm{CS}}$, i.e., f_k is the function obtain by composing $\textsc{And}_n$ with the cheat sheet version of $\textsc{bk-sum}_n$ composed with f_{k-1}.

We prove the claim by induction on k. The induction hypothesis and the base case, $f_1 = \textsc{And}_n \circ \textsc{Bkk}_{\mathrm{CS}}$, are presented below, where N_k is the input size of the function f_k.

Induction hypothesis (f_k)

- $N_k = O(n^{25^k})$
- $Q(f_k) = \widetilde{\Omega}(n^{1.5k+1})$
- $C_0(f_k) = \widetilde{O}(n^k)$
- $C_1(f_k) = \widetilde{O}(n^{k+1})$
- $\mathrm{UC}(f_k) = \widetilde{O}(n^{k+1})$

Base case ($f_1 = \textsc{And}_n \circ \textsc{Bkk}_{\mathrm{CS}}$)

- $N_1 = O(n^{25})$
- $Q(f_1) = \widetilde{\Omega}(n^{2.5})$
- $C_0(f_1) = \widetilde{O}(n)$
- $C_1(f_1) = \widetilde{O}(n^2)$
- $\mathrm{UC}(f_1) = \widetilde{O}(n^2)$

Let us first compute the complexities of $f_1 = \textsc{And}_n \circ \textsc{Bkk}_{\text{CS}}$ and verify that they are consistent with the induction hypothesis. First note that the input size of Bkk is n^2, and thus the input size of $\textsc{Bkk}_{\text{CS}}$ is $O(n^{24})$ (from Lemma 4.2), and hence the input size of $f_1 = \textsc{And}_n \circ \textsc{Bkk}_{\text{CS}}$ is $O(n^{25})$. We have $Q(f_1) = \widetilde{\Omega}(n^{2.5})$ since $Q(f_1) = \Omega(\sqrt{n}Q(\textsc{Bkk}_{\text{CS}})) = \widetilde{\Omega}(n^{2.5})$. The other inequalities follow straightforwardly from Lemma 3.1 and Lemma 6.1. We have $C_0(f_1) = \widetilde{O}(n)$ since $C_0(\textsc{And}_n \circ \textsc{Bkk}_{\text{CS}}) \leq C_0(\textsc{Bkk}_{\text{CS}}) = \widetilde{O}(n)$. We have $C_1(f_1) = \widetilde{O}(n^2)$ since $C_1(\textsc{And}_n \circ \textsc{Bkk}_{\text{CS}}) \leq nC_1(\textsc{Bkk}_{\text{CS}}) = \widetilde{O}(n^2)$. Lastly, we have $\text{UC}(f_1) = \text{UC}(\textsc{And}_n \circ \textsc{Bkk}_{\text{CS}}) \leq \text{UC}_0(\textsc{Bkk}_{\text{CS}}) + n\text{UC}_1(\textsc{Bkk}_{\text{CS}}) \leq D(\textsc{Bkk}_{\text{CS}}) + \widetilde{O}(nC(\textsc{Bkk})) = \widetilde{O}(D(\textsc{Bkk}) + nC(\textsc{Bkk})) = \widetilde{O}(n^2)$.

We now show that the induction hypothesis for f_k implies the same for f_{k+1}. The input size calculation is identical to that in Theorem 5.1 and hence we do not repeat it. The quantum query complexity of f_{k+1} can be lower bounded as follows:

$$\begin{aligned} Q(f_{k+1}) &= Q(\textsc{And}_n \circ (\textsc{bk-sum}_n \circ f_k)_{\text{CS}}) = \Omega(\sqrt{n}Q((\textsc{bk-sum}_n \circ f_k)_{\text{CS}})) \\ &= \Omega(\sqrt{n}Q(\textsc{bk-sum}_n \circ f_k)) = \Omega(n^{1.5}Q(f_k)) = \widetilde{\Omega}(n^{1.5(k+1)+1}), \end{aligned}$$

where we used Lemma 3.1, Lemma 4.2, Lemma 6.2, and Lemma 6.3 to compute the relevant measures.

Similarly we have

$$\begin{aligned} C_0(f_{k+1}) &= C_0(\textsc{And}_n \circ (\textsc{bk-sum}_n \circ f_k)_{\text{CS}}) \leq C_0((\textsc{bk-sum}_n \circ f_k)_{\text{CS}}) \\ &= \widetilde{O}(C(\textsc{bk-sum}_n \circ f_k)) = \widetilde{O}(nC_0(f_k) + C_1(f_k)) = \widetilde{O}(n^{k+1}) \qquad \text{and} \\ C_1(f_{k+1}) &= C_1(\textsc{And}_n \circ (\textsc{bk-sum}_n \circ f_k)_{\text{CS}}) \leq nC_1((\textsc{bk-sum}_n \circ f_k)_{\text{CS}}) \\ &= \widetilde{O}(nC(\textsc{bk-sum}_n \circ f_k)) = \widetilde{O}(n^2C_0(f_k) + nC_1(f_k)) = \widetilde{O}(n^{k+2}). \end{aligned}$$

Finally, using Lemma 3.1, Lemma 4.2, Lemma 6.2, and Lemma 6.3 again we have

$$\begin{aligned} \text{UC}(f_{k+1}) &= \text{UC}(\textsc{And}_n \circ (\textsc{bk-sum}_n \circ f_k)_{\text{CS}}) \\ &= O\left(\max\left\{\text{UC}_0((\textsc{bk-sum}_n \circ f_k)_{\text{CS}}), n\text{UC}_1((\textsc{bk-sum}_n \circ f_k)_{\text{CS}})\right\}\right) \\ &= \widetilde{O}\left(\max\left\{\text{UC}(\textsc{bk-sum}_n \circ f_k), nC(\textsc{bk-sum}_n \circ f_k)\right\}\right) \\ &= \widetilde{O}\left(\max\left\{n\text{UC}(f_k), n^2C_0(f_k) + nC_1(f_k)\right\}\right) = \widetilde{O}(n^{k+2}). \end{aligned}$$

This completes the induction and establishes the first part of the theorem. Using a similar argument in Theorem 5.1, this yields a function with $Q(f) \geq \text{UC}(f)^{1.5-o(1)}$. ◀

Finally, this establishes Theorem 1.4.

▶ **Theorem 1.4.** *There exists a total function f with $Q(f) \geq \text{UC}(f)^{1.5-o(1)}$.*

Acknowledgements. R. K. thanks Shalev Ben-David for several helpful conversations and comments on a preliminary draft.

This work is supported by ARO grant number W911NF-12-1-0486, the European Commission FET-Proactive project QALGO, ERC Advanced Grant MQC and Latvian State Research programme NexIT project No 1.

References

1 Scott Aaronson. Quantum certificate complexity. *SIAM Journal on Computing*, 35(4):804–824, 2006.

2 Scott Aaronson. Lower bounds for local search by quantum arguments. *Journal of Computer and System Sciences*, 74(3):313–332, 2008.

3 Scott Aaronson, Shalev Ben-David, and Robin Kothari. Separations in query complexity using cheat sheets. In *Proceedings of the 48th ACM Symposium on Theory of Computing (STOC 2016)*, to appear. arXiv:arXiv:1511.01937.

4 Alfred V. Aho, Jeffrey D. Ullman, and Mihalis Yannakakis. On notions of information transfer in VLSI circuits. In *Proceedings of the 15th ACM Symposium on Theory of Computing (STOC 1983)*, pages 133–139, 1983. doi:10.1145/800061.808742.

5 Charles H. Bennett, Ethan Bernstein, Gilles Brassard, and Umesh Vazirani. Strengths and weaknesses of quantum computing. *SIAM Journal on Computing (special issue on quantum computing)*, 26:1510–1523, 1997.

6 Harry Buhrman and Ronald de Wolf. Complexity measures and decision tree complexity: a survey. *Theoretical Computer Science*, 288(1):21–43, 2002. doi:10.1016/S0304-3975(01)00144-X.

7 Justin Gilmer, Michael Saks, and Srikanth Srinivasan. Composition limits and separating examples for some Boolean function complexity measures. In *Proceedings of 2013 IEEE Conference on Computational Complexity (CCC 2013)*, pages 185–196, June 2013. doi:10.1109/CCC.2013.27.

8 Mika Göös. Lower bounds for clique vs. independent set. In *Proceedings of the 56th IEEE Symposium on Foundations of Computer Science (FOCS 2015)*, pages 1066–1076, 2015. doi:10.1109/FOCS.2015.69.

9 Mika Göös, T.S. Jayram, Toniann Pitassi, and Thomas Watson. Randomized communication vs. partition number. *Electronic Colloquium on Computational Complexity (ECCC) TR15-169*, 2015.

10 Mika Göös, Toniann Pitassi, and Thomas Watson. Deterministic communication vs. partition number. In *Proceedings of the 56th IEEE Symposium on Foundations of Computer Science (FOCS 2015)*, pages 1077–1088, 2015. doi:10.1109/FOCS.2015.70.

11 Lov K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the 28th ACM Symposium on Theory of Computing (STOC 1996)*, pages 212–219, 1996. doi:10.1145/237814.237866.

12 Peter Høyer, Troy Lee, and Robert Špalek. Negative weights make adversaries stronger. In *Proceedings of the 39th ACM Symposium on Theory of Computing (STOC 2007)*, pages 526–535, 2007. doi:10.1145/1250790.1250867.

13 Rahul Jain and Hartmut Klauck. The partition bound for classical communication complexity and query complexity. In *Proceedings of the 2010 IEEE 25th Annual Conference on Computational Complexity*, CCC'10, pages 247–258, 2010. doi:10.1109/CCC.2010.31.

14 Rahul Jain, Troy Lee, and Nisheeth K. Vishnoi. A quadratically tight partition bound for classical communication complexity and query complexity. *arXiv preprint* arXiv:1401.4512, 2014.

15 Stasys Jukna. *Boolean Function Complexity: Advances and Frontiers*. Algorithms and Combinatorics. Springer, 2012.

16 Jedrzej Kaniewski, Troy Lee, and Ronald de Wolf. Query complexity in expectation. In *Automata, Languages, and Programming*, volume 9134 of *Lecture Notes in Computer Science*, pages 761–772. Springer Berlin Heidelberg, 2015. doi:10.1007/978-3-662-47672-7_62.

17 Robin Kothari, David Racicot-Desloges, and Miklos Santha. Separating Decision Tree Complexity from Subcube Partition Complexity. In *Approximation, Randomization, and*

Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2015), volume 40 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 915–930, 2015. doi:10.4230/LIPIcs.APPROX-RANDOM.2015.915.

18 Eyal Kushilevitz, Nathan Linial, and Rafail Ostrovsky. The linear-array conjecture in communication complexity is false. *Combinatorica*, 19(2):241–254, 1999. doi:10.1007/s004930050054.

19 Eyal Kushilevitz and Noam Nisan. *Communication Complexity*. Cambridge University Press, 2006.

20 Sophie Laplante and Frédéric Magniez. Lower bounds for randomized and quantum query complexity using Kolmogorov arguments. *SIAM Journal on Computing*, 38(1):46–62, 2008.

21 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 2011)*, pages 344–353, 2011. arXiv:arXiv:1011.3020, doi:10.1109/FOCS.2011.75.

22 Troy Lee and Adi Shraibman. Lower bounds in communication complexity. *Foundations and Trends® in Theoretical Computer Science*, 3(4):263–399, 2007. doi:10.1561/0400000040.

23 Ashley Montanaro. A composition theorem for decision tree complexity. *Chicago Journal of Theoretical Computer Science*, 2014(6), July 2014.

24 Noam Nisan. CREW PRAMs and decision trees. *SIAM Journal on Computing*, 20(6):999–1007, 1991. doi:10.1137/0220062.

25 Noam Nisan and Mario Szegedy. On the degree of Boolean functions as real polynomials. *Computational Complexity*, 15(4):557–565, 1995.

26 Ben W. Reichardt. Reflections for quantum query algorithms. In *Proceedings of the 22nd ACM-SIAM Symposium on Discrete Algorithms (SODA 2011)*, SODA'11, pages 560–569, 2011. URL: http://dl.acm.org/citation.cfm?id=2133036.2133080.

27 Petr Savický. On determinism versus unambiquous nondeterminism for decision trees. *ECCC*, TR02-009, 2002.

28 Robert Špalek and Mario Szegedy. All quantum adversary methods are equivalent. *Theory of Computing*, 2(1):1–18, 2006.

29 Avishay Tal. Properties and applications of Boolean function composition. In *Proceedings of the 4th Conference on Innovations in Theoretical Computer Science*, ITCS'13, pages 441–454, 2013. doi:10.1145/2422436.2422485.

30 Andrew Chi-Chih Yao. Some complexity questions related to distributive computing. In *Proceedings of the 11th ACM Symposium on Theory of Computing*, STOC'79, pages 209–213, 1979. doi:10.1145/800135.804414.

A Composition Theorem for Conical Juntas

Mika Göös[*1] **and T. S. Jayram**[2]

1 **University of Toronto, Toronto, Canada**
mika.goos@mail.utoronto.ca
2 **IBM Research – Almaden, San Jose, USA**
jayram@us.ibm.com

Abstract

We describe a general method of proving degree lower bounds for *conical juntas* (nonnegative combinations of conjunctions) that compute recursively defined boolean functions. Such lower bounds are known to carry over to communication complexity. We give two applications:

- **AND-OR trees.** We show a near-optimal $\tilde{\Omega}(n^{0.753\cdots})$ randomised communication lower bound for the recursive NAND function (a.k.a. AND-OR tree). This answers an open question posed by Beame and Lawry [6, 23].
- **Majority trees.** We show an $\Omega(2.59^k)$ randomised communication lower bound for the 3-majority tree of height k. This improves over the state-of-the-art already in the context of randomised *decision tree* complexity.

1998 ACM Subject Classification F.1.3 Complexity Measures and Classes

Keywords and phrases Composition theorems, conical juntas

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.5

1 Conical Juntas?

Conical juntas are nonnegative linear combinations of conjunctions. Here are two examples, one computing the two-bit OR function $\mathsf{OR}\colon \{0,1\}^2 \to \{0,1\}$ and another computing the three-bit majority function $\mathsf{Maj}_3\colon \{0,1\}^3 \to \{0,1\}$:

$$\begin{aligned} h_1(x) &= \tfrac{1}{2}x_1 + \tfrac{1}{2}x_2 + \tfrac{1}{2}\bar{x}_1x_2 + \tfrac{1}{2}x_1\bar{x}_2, \\ h_2(y) &= \tfrac{1}{3}y_1y_2 + \tfrac{1}{3}y_2y_3 + \tfrac{1}{3}y_1y_3 + \tfrac{2}{3}\bar{y}_1y_2y_3 + \tfrac{2}{3}y_1\bar{y}_2y_3 + \tfrac{2}{3}y_1y_2\bar{y}_3. \end{aligned} \tag{1}$$

The purpose of this work is to prove lower bounds on the *degree* $\deg(h)$ (maximum width of a conjunction in h) of any conical junta h that computes – even approximately – a given boolean function $f\colon \{0,1\}^n \to \{0,1\}$. More precisely, we study the ϵ-*approximate* conical junta degree of f, denoted $\deg_\epsilon(f)$, that is defined as the minimum degree of a conical junta h satisfying

$$\forall x: \quad |h(x) - f(x)| \;\leq\; \epsilon.$$

Communication complexity connection. A major motivation for studying conical junta degree comes from the works [10, 13, 24] that connect conical juntas with *nonnegative rank*, a basic measure in communication complexity. Roughly speaking, lower bounds on approximate conical junta degree of f can be translated into lower bounds on the approximate nonnegative rank of a certain two-party "lift" of f, and therefore into lower bounds against randomised protocols.

[*] Work done while at IBM Research Almaden.

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 5; pp. 5:1–5:16
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Related models. Conical juntas have been studied under such names as the (one-sided) *partition bound* for query complexity [15] and *query complexity in expectation* [20]. Another closely related model is that of *randomised subcube partitions* [11, 17, 21]. Moreover, if we restrict the coefficients in a conical junta to be 0-1, we obtain the model of subcube partitions a.k.a. unambiguous DNFs [30, 7, 12, 14, 21].

2 Our Results

Our main technical result is a *Composition Theorem* that makes it easy to prove conical junta degree lower bounds for functions that are defined from simpler functions via composition. If f and g are boolean functions on n and m bits, respectively, their *composition* $f \circ g^n$ is the function on nm bits that maps an input $x = (x_1, \dots, x_n) \in (\{0,1\}^m)^n$ to the output

$$(f \circ g^n)(x) \;\coloneqq\; f(g(x_1), \dots, g(x_n)).$$

Define also $f^{\circ k} \coloneqq f \circ (f^{\circ(k-1)})^n$ where $f^{\circ 1} \coloneqq f$. The exact statement of the Composition Theorem is deferred to Section 4 as it is somewhat technical. It is phrased in terms of *dual solutions* (or *certificates*) to a linear program that captures a certain *average* version of conical junta degree (defined in Section 3). The theorem splits the task of proving lower bounds into two steps: we first need to find dual certificates for f and g (e.g., by solving an LP, either by inspection, or by using a computer), and then we can let the Composition Theorem construct a dual certificate for $f \circ g^n$ in a black-box fashion. We note that similar LP-based approaches have been extremely popular in analysing the degree of multivariate polynomials (see [31, 32, 9] for recent examples) – in short, this work develops such a framework for conical juntas, a nonnegative analogue of multivariate polynomials.

Setting these technical matters aside for a moment, let us illustrate the power the Composition Theorem by looking at some of its consequences.

2.1 Query complexity

We give applications for two well-studied recursively defined boolean functions; see Figure 1.

▶ **Theorem 2.1.** $\deg_\epsilon(\mathsf{NAND}^{\circ k}) \geq \Omega(n^{0.753\dots})$ *for all* $\epsilon \leq 1/n$ *where* $n \coloneqq 2^k$.

▶ **Theorem 2.2.** $\deg_\epsilon(\mathsf{Maj}_3^{\circ k}) \geq \Omega(2.59\dots^k)$ *for all* $\epsilon \leq 1/n$ *where* $n \coloneqq 3^k$.

Discussion of Theorem 2.1. The function $\mathsf{NAND}^{\circ k}$ is computed by a height-k binary tree consisting of NAND gates (a.k.a. AND-OR tree). A classical result [28, 29] states that any randomised decision tree needs to query $\Omega(n^{0.753\dots})$ (here $0.753\dots = \log(1+\sqrt{33})-2$) many input bits in order to compute $\mathsf{NAND}^{\circ k}$ with high probability. This matches an upper bound due to Snir [33]. Our Theorem 2.1 shows that the same lower bound holds even for conical juntas that approximate $\mathsf{NAND}^{\circ k}$ sufficiently well. This is a qualitative strengthening of the classical results since conical juntas are relaxations of decision trees. Indeed, a randomised decision tree of depth d that computes a function f to within error $\epsilon > 0$ can be converted into a degree-d ϵ-approximate conical junta for f – the reason is the same as for multivariate polynomials [8, Theorem 15]. Speaking of polynomials, Theorem 2.1 should be compared with the fact that the approximate polynomial degree of $\mathsf{NAND}^{\circ k}$ is only $O(\sqrt{n})$ (and this upper bound holds even for quantum algorithms [5]).

Note: A caveat with Theorems 2.1–2.2 is that we only know how to prove them for $\epsilon \leq 1/n$. By contrast, one usually takes $\epsilon = 1/3$ when studying decision trees, and this is

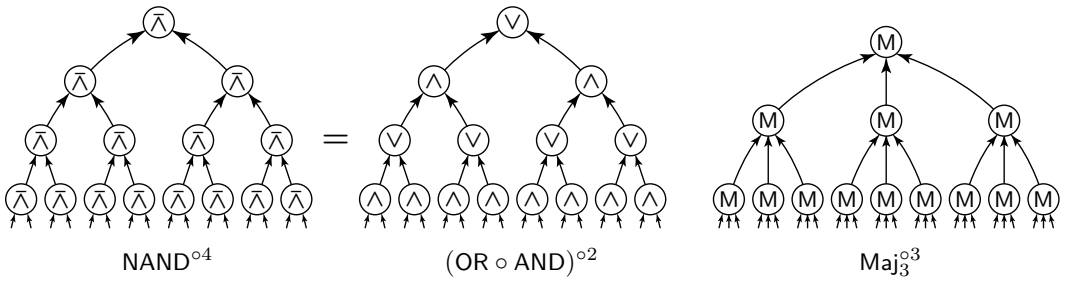


Figure 1 Examples of recursively defined boolean functions studied in this work.

well-known to be w.l.o.g., because the error can be reduced below any $\epsilon < 1/3$ with only a factor $O(\log(1/\epsilon))$ increase in query complexity. Interestingly, for conical juntas, it is known [13] that ϵ cannot always be efficiently reduced: for any constants $\epsilon > \delta > 0$ there exists a *partial* function f with $\deg_\epsilon(f) = 1$ but $\deg_\delta(f) \geq \Omega(n)$. For *total* functions, it is still open whether efficient error reduction is possible (standard techniques [8] at least show that $\deg_\epsilon(f)$ is polynomially related to $\deg_0(f)$). In any case, Theorems 2.1–2.2 do indeed imply lower bounds for randomised decision trees with error $\epsilon = 1/3$: we simply have to reduce the error below $1/n$ first and only then convert the decision tree into a conical junta. This incurs a factor $\Theta(\log n)$ loss in the value of the lower bound.

Discussion of Theorem 2.2. For the reasons discussed above, Theorem 2.2 implies a lower bound of $\tilde{\Omega}(2.59\ldots^k) \geq \Omega(2.59^k)$ (here $2.59\ldots = \sqrt[3]{35/2}$, and the $\tilde{\Omega}$-notation hides polylog(n) factors) for the randomised query complexity of the recursive majority function $\mathsf{Maj}_3^{\circ k}$. This slightly improves over the previous bound of $\Omega(2.57^k)$ that is the culmination of the line of work [19, 22, 25, 27] wielding information theoretic tools. For comparison, a randomised zero-error decision tree of cost $O(2.65^k)$ is known [27]. Even though our quantitative improvement in Theorem 2.2 is modest, the theorem nevertheless suggests that our new techniques are rather powerful: they are already competitive with highly optimised prior work, especially [27].

2.2 Communication complexity

Using the machinery of [13] we can now translate Theorems 2.1–2.2 into analogous communication results. The translation incurs some polylog(n) factor loss in parameters, which is suppressed by the $\tilde{\Omega}$-notation used below. Here $\mathsf{BPP}^{\mathsf{cc}}(F)$ stands for the bounded-error communication complexity of F under a worst-case Alice–Bob bipartition of the input bits. For our functions, we may take the bipartition to be such that Alice gets the first bit of every bottom gate and Bob gets the rest.

▶ **Theorem 2.3.** $\mathsf{BPP}^{\mathsf{cc}}(\mathsf{NAND}^{\circ k}) \geq \tilde{\Omega}(n^{0.753\ldots})$.

▶ **Theorem 2.4.** $\mathsf{BPP}^{\mathsf{cc}}(\mathsf{Maj}_3^{\circ k}) \geq \Omega(2.59^k)$.

Discussion of Theorem 2.3. The question of proving a lower bound for the randomised communication complexity of the balanced alternating AND-OR tree (with fan-in 2 gates next to the inputs) having n leaves was first posed by Beame and Lawry [6, 23] to the best of our knowledge. They were interested in matching the randomised query complexity bound, towards separating randomized communication complexity from both nondeterministic and

co-nondeterministic communication complexity. Two independent works [18, 26] (building on [19]) arrived at a lower bound of $\Omega(n/2^{O(k)})$ (or slightly worse $\Omega(n/k^{O(k)})$ in [18]) for the randomised communication complexity of any height-k unbounded fan-in alternating AND-OR tree (with fan-in 2 gates next to the inputs). While this lower bound is tight when $k = O(1)$, the bound becomes trivial in the setting of Theorem 2.3 where $k = \log n$. This shortcoming was partially addressed by [16] who showed, via a reduction from set-disjointness, a lower bound of $\Omega(\sqrt{n})$ for such AND-OR trees, independently of the height. Our Theorem 2.3 now gives an essentially optimal $\tilde{\Omega}(n^{0.753\cdots})$ bound for the particular case of $\mathsf{NAND}^{\circ k}$. It remains open whether this lower bound holds for *all* AND-OR trees (with the appropriate gates next to the inputs). For *query complexity*, Amano [1] has come close to settling this question, known as the Saks–Wigderson conjecture [28] for the class of read-once formulas (a more general version of the conjecture was recently disproved [4]).

Discussion of Theorem 2.4. The function $\mathsf{Maj}_3^{\circ k}$ has not been studied in communication complexity previously – after all, even its randomised query complexity is not yet completely understood.

3 Definitions and Examples

We write $h = \sum w_C C$ for a generic conical junta, where the sum ranges over different conjunctions of literals $C\colon \{0,1\}^n \to \{0,1\}$ and $w_C \geq 0$ for each C. Note that $h\colon \{0,1\}^n \to \mathbb{R}_{\geq 0}$. Let $|C|$ denote the width of a conjunction C, i.e., the number of literals in C. The degree of h, denoted $\deg(h)$, is defined as the maximum width of a conjunction C with $w_C > 0$. Here, it is helpful to work with a more robust notion of degree that we call *average degree*. The *average degree* of h, denoted $\mathrm{adeg}(h)$, is defined as the maximum over all inputs x of

$$\mathrm{adeg}_x(h) \coloneqq \sum w_C |C| C(x) = \sum w_C \,\mathrm{adeg}_x(C).$$

In particular, $\mathrm{adeg}(h) \leq \deg(h)$ in the natural setting where $h(x) \leq 1$ for all x. Our definition of average degree is in perfect analogy to the usual definition of cost for randomised zero-error decision trees, namely, charging for the *expected* number of queries made on a given input. Indeed, it is not hard to see that any zero-error decision tree of cost d gives rise to a conical junta of average degree d computing exactly the same boolean function as the decision tree.

For a boolean function $f\colon \{0,1\}^n \to \{0,1\}$ we define

- *Degree:* $\deg(f)$ is the minimum $\deg(h)$ over all conical juntas h computing f.
- *Average degree:* $\mathrm{adeg}(f)$ is the minimum $\mathrm{adeg}(h)$ over all conical juntas h computing f.
- *Approximate degree:* $\deg_\epsilon(f)$ is the minimum $\deg(h)$ over all conical juntas h that compute f to within error ϵ, i.e., $h(x) \in f(x) \pm \epsilon$ for all x.

3.1 Tame examples

For our conical juntas h_1 and h_2 from (1), we have $\mathrm{adeg}(h_1) = \mathrm{adeg}_{10}(h_1) = 3/2 < 2 = \deg(h_1)$ and $\mathrm{adeg}(h_2) = \mathrm{adeg}_{110}(h_2) = 8/3 < 3 = \deg(h_2)$. In fact, h_1 and h_2 are optimal:

$$\mathrm{adeg}(\mathsf{OR}) = 3/2 \qquad \text{and} \qquad \mathrm{adeg}(\mathsf{Maj}_3) = 8/3.$$

This can be seen by solving an LP whose value is $\mathrm{adeg}(f)$, as is discussed shortly. Note that our degree measures are inherently *one-sided*: f and its negation $\neg f$ need not have the same

degree. For example, we have $\mathrm{adeg}(\neg\mathsf{OR}) = 2$ (observe that $\bar{x}_1\bar{x}_2$ is the only conical junta for $\neg\mathsf{OR}$) even though $\mathrm{adeg}(\mathsf{OR}) = 3/2$. (More dramatic gaps can be demonstrated using variations of a function introduced in [14].) By contrast, Maj_3 is *self-dual*, $\neg\mathsf{Maj}_3(x_1, x_2, x_3) = \mathsf{Maj}_3(\neg x_1, \neg x_2, \neg x_3)$, so we automatically have $\mathrm{adeg}(\mathsf{Maj}_3) = \mathrm{adeg}(\neg\mathsf{Maj}_3)$.

3.2 A wild example!

What is the average degree of $\mathsf{OR} \circ \mathsf{Maj}_3^2$? We can obtain a conical junta for this function starting with the optimal conical juntas $h_1(x)$, $h_2(y)$, $\bar{h}_2(y) \coloneqq h_2(\bar{y}_1, \bar{y}_2, \bar{y}_3)$ computing OR, Maj_3, $\neg\mathsf{Maj}_3$, respectively, as follows: Let $z^1 = (z^1_1, z^1_2, z^1_3)$ and $z^2 = (z^2_1, z^2_2, z^2_3)$ be fresh variables. Start with $h_1(x)$ and replace every positive literal x_i by $h_2(z^i)$ and every negative literal $\bar{x}_i$ by $\bar{h}_2(z^i)$. This construction shows that

$$\mathrm{adeg}(\mathsf{OR} \circ \mathsf{Maj}_3^2) \;\le\; 3/2 \cdot 8/3 \;=\; 4.$$

It would be natural to conjecture that this is tight – *but this conjecture is false!* There is in fact a more effective conical junta of average degree only $47/12 \approx 3.92$. An analogous phenomenon is well-known in the context of zero-error decision trees: so-called *directional* decision trees need not be optimal for composed functions [28, 34, 2].

What of it? This example shows that we cannot hope for a perfect composition theorem for average degree that would determine $\mathrm{adeg}(f \circ g^n)$ solely in terms of $\mathrm{adeg}(f)$, $\mathrm{adeg}(g)$, and $\mathrm{adeg}(\neg g)$, even assuming $\mathrm{adeg}(g) = \mathrm{adeg}(\neg g)$. Consequently, for our LP-based Composition Theorem, we will have to introduce *some* technical assumptions: to enable the construction of a dual certificate for $\mathrm{adeg}(f \circ g^n)$, we assume we have dual certificates *of a special form* for $\mathrm{adeg}(f)$, $\mathrm{adeg}(g)$, $\mathrm{adeg}(\neg g)$. The rest of this section develops our LP formalism for average degree.

3.3 Generalised input costs

Let us first generalise the definition of $\mathrm{adeg}(h)$ by allowing arbitrary costs $b_0, b_1 \ge 0$ to be assigned to reading the input bits. That is, for a conjunction C, we set $|C|_{b_0,b_1} \coloneqq b_0 \cdot (\#\text{ of 0's read by } C) + b_1 \cdot (\#\text{ of 1's read by } C)$. In particular, $|C|_{1,1} = |C|$. Then $\mathrm{adeg}(h; b_0, b_1)$ is defined as the maximum over all inputs x of

$$\mathrm{adeg}_x(h; b_0, b_1) \;\coloneqq\; \sum w_C |C|_{b_0,b_1} C(x) \;=\; \sum w_C \,\mathrm{adeg}_x(C; b_0, b_1).$$

We also introduce some "distributional" notation: for a distribution D_1 over $f^{-1}(1)$ we let

$$\mathrm{adeg}_{D_1}(h; b_0, b_1) \;\coloneqq\; \mathop{\mathbf{E}}_{x \sim D_1} \big[\mathrm{adeg}_x(h; b_0, b_1)\big].$$

For a boolean function $f\colon \{0,1\}^n \to \{0,1\}$ we define

- $\mathrm{adeg}(f; b_0, b_1)$ is the minimum of $\mathrm{adeg}(h; b_0, b_1)$ over all conical juntas h computing f.
- $\mathrm{adeg}_{D_1}(f; b_0, b_1)$ is the minimum of $\mathrm{adeg}_{D_1}(h; b_0, b_1)$ over all conical juntas h computing f.

It is clear that $\mathrm{adeg}(f; b_0, b_1) \ge \mathrm{adeg}_{D_1}(f; b_0, b_1)$ for all distributions D_1. (In fact, it can be shown using the minimax theorem that this inequality can be turned into an equality if we maximise over D_1 on the right hand side – however, we do not use this fact.)

3.4 An LP for average degree

We formulate $\mathrm{adeg}_{D_1}(f; b_0, b_1)$ as the optimum value of an LP – here the data f, D_1, b_0, b_1, is thought of as fixed. We have a nonnegative variable $w_C \geq 0$ for each of the 3^n possible conjunctions $C\colon \{0,1\}^n \to \{0,1\}$. Here is the LP:

$$\begin{array}{rlll} \min & \mathrm{adeg}_{D_1}\left(\sum w_C C; b_0, b_1\right) & & \\ \textit{subject to} & \sum w_C C(x) = f(x), & \forall x & \text{(Primal)} \\ & w_C \geq 0, & \forall C & \end{array}$$

Here is the LP dual; the free variables are packaged into a function $\Psi\colon \{0,1\}^n \to \mathbb{R}$.

$$\begin{array}{rlll} \max & \langle \Psi, f \rangle & & \\ \textit{subject to} & \langle \Psi, C \rangle \leq \mathrm{adeg}_{D_1}(C; b_0, b_1), & \forall C & \text{(Dual)} \\ & \Psi(x) \in \mathbb{R}, & \forall x & \end{array}$$

Since we are interested in proving lower bounds on average degree, we are only going to need the "weak" form of LP duality: Suppose $h = \sum w_C C$ is an optimal solution to (Primal). Then any solution Ψ that is feasible for (Dual) witnesses a lower bound on $\mathrm{adeg}(f; b_0, b_1)$ like so:

$$\begin{aligned} \mathrm{adeg}(f; b_0, b_1) &\geq \mathrm{adeg}_{D_1}(f; b_0, b_1) \\ &= \mathrm{adeg}_{D_1}(h; b_0, b_1) \\ &= \textstyle\sum w_C \,\mathrm{adeg}_{D_1}(C; b_0, b_1) \\ &\geq \textstyle\sum w_C \langle \Psi, C \rangle \\ &= \textstyle\langle \Psi, \sum w_C C \rangle \\ &= \langle \Psi, f \rangle. \end{aligned} \tag{2}$$

4 Statement of the Composition Theorem

We start by defining an $(a_0, a_1; b_0, b_1)$-*certificate* for f as a special collection of certificates witnessing

$$\begin{aligned} \mathrm{adeg}(f; b_0, b_1) &\geq a_1, \\ \mathrm{adeg}(\neg f; b_0, b_1) &\geq a_0. \end{aligned} \tag{3}$$

▶ **Definition 4.1.** Call a function $\Psi\colon \{0,1\}^n \to \mathbb{R}$ *balanced* if $\sum_x \Psi(x) = 0$, and also write $X_{\geq 0} \coloneqq \max\{X, 0\}$ for short. An $(a_0, a_1; b_0, b_1)$-*certificate* for a function $f\colon \{0,1\}^n \to \{0,1\}$ consists of four balanced functions $\{\Psi_v, \hat{\Psi}_v\}_{v=0,1}$ mapping $\{0,1\}^n \to \mathbb{R}$ such that the following hold.

- **Special form:** Functions Ψ_0 and Ψ_1 have the form

$$\Psi_v = a_v(D_v - D_{1-v}), \tag{4}$$

where D_v is a distribution over $f^{-1}(v)$. Moreover, $\hat{\Psi}_v$ is supported on $f^{-1}(v)$.

- **Feasibility:** For all conjunctions C and $v \in \{0,1\}$,

$$\langle \Psi_v, C\rangle_{\geq 0} + \langle \hat{\Psi}_v, C\rangle \;\leq\; \mathrm{adeg}_{D_v}(C; b_0, b_1). \tag{5}$$

▶ **Theorem 4.2** (Composition Theorem). *Suppose f admits an $(a_0, a_1; b_0, b_1)$-certificate and g admits a $(b_0, b_1; 1, 1)$-certificate. Then $f \circ g^n$ admits an $(a_0, a_1; 1, 1)$-certificate.*

Discussion. First, we note that (5) actually packs together two *linear* inequalities; it would be equivalent to require that both $\Psi_v + \hat{\Psi}_v$ and $\hat{\Psi}_v$ are feasible for (Dual), namely that

$$\begin{cases} \langle \Psi_v + \hat{\Psi}_v, C\rangle \;\leq\; \mathrm{adeg}_{D_v}(C; b_0, b_1), \\ \langle \hat{\Psi}_v, C\rangle \;\leq\; \mathrm{adeg}_{D_v}(C; b_0, b_1). \end{cases} \tag{5'}$$

Here $\Psi_1 + \hat{\Psi}_1$ is the main attraction: it witnesses a lower bound of $\langle \Psi_1 + \hat{\Psi}_1, f\rangle = \langle \Psi_1, f\rangle + \langle \hat{\Psi}_1, f\rangle = a_1 + 0 = a_1$ for $\mathrm{adeg}(f; b_0, b_1)$ as promised above (3); similarly, $\Psi_0 + \hat{\Psi}_0$ witnesses the complementary lower bound $\mathrm{adeg}(\neg f; b_0, b_1) \geq a_0$.

The requirement that $\Psi_1 + \hat{\Psi}_1$ must be balanced is perhaps our most critical assumption. We use it to manoeuvre around the counterexample of Section 3.2: we have $\mathrm{adeg}(\mathsf{Maj}_3) = 8/3$, while the best *balanced* solution to (Dual) only witnesses the lower bound $\mathrm{adeg}(\mathsf{Maj}_3) \geq 5/2$ (see also Figure 3). The requirement that $\hat{\Psi}_v$ is feasible for (Dual) is merely a technical assumption that helps us in the upcoming proof (akin to a "strengthened induction hypothesis"); we do not know whether the theorem is true without this condition. Another technical assumption is (4), which allows us to assume that Ψ_1 and Ψ_0 have opposite signs: $\Psi_1 = -a_1/a_0 \cdot \Psi_0$.

Some simple certificates are illustrated in Figures 2–3. Their feasibility can be checked by hand. For more involved functions, certificates can in principle be found via a computer search (using computers is not uncommon even in "lower bounds" research [3]). We will in fact use this approach for $\mathsf{Maj}_3^{\circ k}$ in Section 6.

5 Proof of the Composition Theorem

Let $\{\Psi_v, \hat{\Psi}_v\}_{v=0,1}$ and $\{\Phi_v, \hat{\Phi}_v\}_{v=0,1}$ be the certificates for f and g, respectively. Our goal is to construct a certificate $\{\Upsilon_v, \hat{\Upsilon}_v\}_{v=0,1}$ for $f \circ g^n$. We use the following notation:

$$\underbrace{\Psi_v \;\coloneqq\; a_v(F_v - F_{1-v})}_{\text{given}}, \qquad \underbrace{\Phi_v \;\coloneqq\; b_v(G_v - G_{1-v})}_{\text{given}}, \qquad \underbrace{\Upsilon_v \;\coloneqq\; a_v(D_v - D_{1-v})}_{\text{want to construct}}.$$

By assumption, the distribution F_v is supported on $f^{-1}(v)$ and G_v is supported on $g^{-1}(v)$. We will define D_v to be supported on $(f \circ g^n)^{-1}(v)$.

5.1 Construction

Lifts. Let $\Gamma\colon \{0,1\}^n \to \mathbb{R}$ and suppose that for each $y \in \{0,1\}^n$ we have a function $H_y\colon \{0,1\}^{mn} \to \mathbb{R}$ supported on $(g^n)^{-1}(y) = g^{-1}(y_1) \times \cdots \times g^{-1}(y_n)$. The *lift* of Γ by H is

$$\Gamma^H \;\coloneqq\; \textstyle\sum_{y \in \{0,1\}^n} \Gamma(y) \cdot H_y.$$

In particular, if Γ and the H_y's are probability distributions, so is Γ^H. Note also that if Γ is supported on $f^{-1}(v)$, then Γ^H is supported on $(f \circ g^n)^{-1}(v)$.

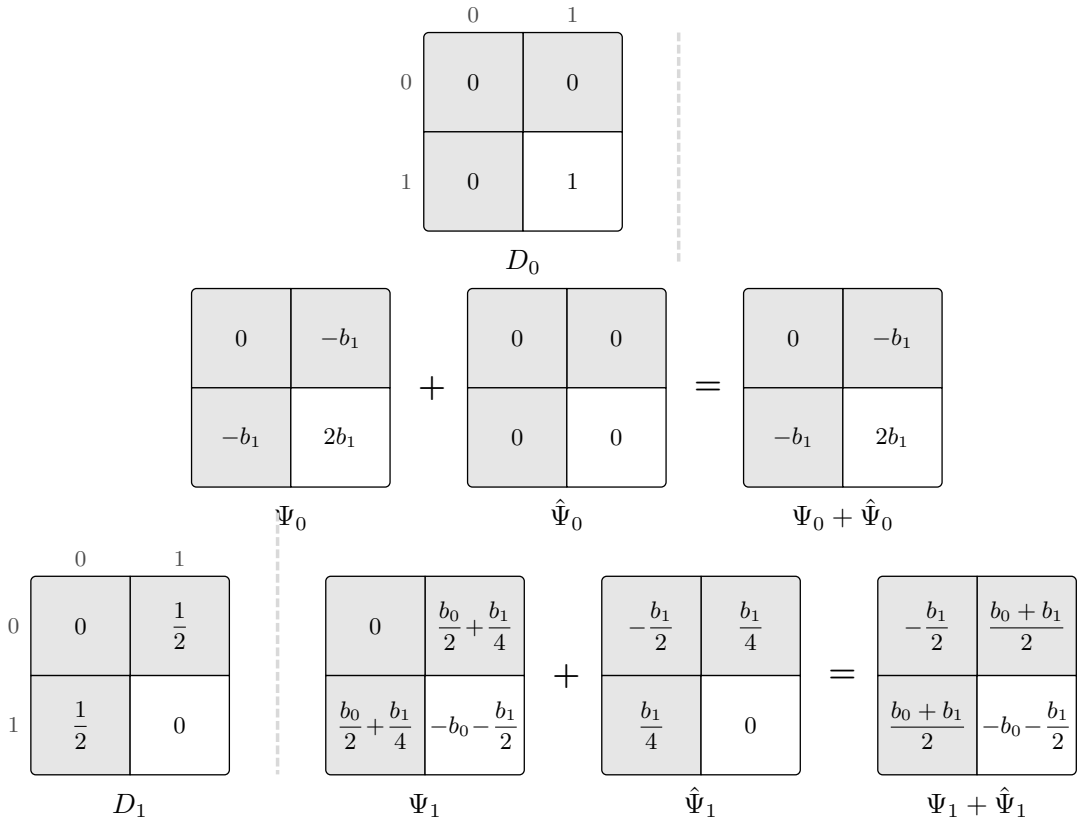


Figure 2 A $(2b_1, b_0+\frac{1}{2}b_1; b_0, b_1)$-certificate for $\mathsf{NAND}\colon \{0,1\}^2 \to \{0,1\}$ that is valid for all $b_0, b_1 \geq 0$. The 1-inputs $\mathsf{NAND}^{-1}(1)$ are highlighted in gray. For feasibility, there are 6 equivalence classes (see Section 6.2) of conjunctions to check: $\{**, *0, *1, 00, 10, 11\}$.

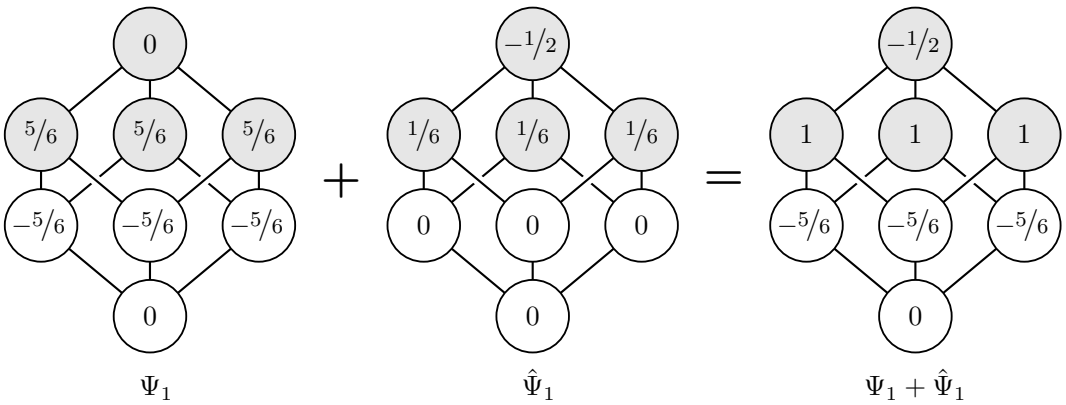


Figure 3 A $(\frac{5}{2}, \frac{5}{2}; 1, 1)$-certificate for $\mathsf{Maj}_3\colon \{0,1\}^3 \to \{0,1\}$. The 1-inputs $\mathsf{Maj}_3^{-1}(1)$ are highlighted in gray. Only $\Psi_1, \hat{\Psi}_1$ are shown as $\Psi_0, \hat{\Psi}_0$ are defined via self-duality. Here D_v is uniform on inputs of Hamming weight $v+1$. For feasibility, there are 10 equivalence classes of conjunctions to check: $\{***, **1, **0, *00, *10, *11, 000, 100, 110, 111\}$. Note that for any $\alpha \geq 0$, we can obtain an $(\frac{5}{2}\alpha, \frac{5}{2}\alpha; \alpha, \alpha)$-certificate by simply scaling the functions $\Psi_v, \hat{\Psi}_v$ by α.

New certificate. Write $G_y \coloneqq G_{y_1} \times \cdots \times G_{y_n}$ for the canonical product distribution on $(g^n)^{-1}(y)$. We also need a modified version of G_y, denoted $(G^{\xleftarrow{i}\hat{\Phi}})_y$ where $i \in [n]$, that has a copy of $\hat{\Phi}_{y_i}$ in place of G_{y_i}; more formally

$$(G^{\xleftarrow{i}\hat{\Phi}})_y(x) \;\coloneqq\; \hat{\Phi}_{y_i}(x_i) \cdot \textstyle\prod_{j\neq i} G_{y_j}(x_j).$$

Note that $(G^{\xleftarrow{i}\hat{\Phi}})_y$ is a balanced function supported on $(g^n)^{-1}(y)$.

We now define $\{\Upsilon_v, \hat{\Upsilon}_v\}_{v=0,1}$ by

$$\begin{aligned} \Upsilon_v &\coloneqq \Psi_v^G, \\ \hat{\Upsilon}_v &\coloneqq \hat{\Psi}_v^G + \textstyle\sum_{i=1}^n F_v^{G^{\xleftarrow{i}\hat{\Phi}}}. \end{aligned} \tag{6}$$

Since $\Psi_v^G = a_v(F_v^G - F_{1-v}^G)$, we have $D_v = F_v^G$. It is also easy to check that $\hat{\Upsilon}_v$ is a balanced function supported on $(f \circ g^n)^{-1}(v)$. Hence $\{\Upsilon_v, \hat{\Upsilon}_v\}_{v=0,1}$ is of the special form required of an $(a_0, a_1; 1, 1)$-certificate for $f \circ g^n$. The interesting part is to verify the feasibility condition (5).

5.2 Feasibility

Fix a conjunction C in the domain of $f \circ g^n$. Our goal is to show

$$\langle \Psi_v^G, C\rangle_{\geq 0} + \langle \hat{\Psi}_v^G + \textstyle\sum_i F_v^{G^{\xleftarrow{i}\hat{\Phi}}}, C\rangle \;\leq\; \mathrm{adeg}_{D_v}(C). \tag{7}$$

Extracting a conical junta from C. Our analysis will be centered around a conical junta $h(y)$, defined below, that computes the acceptance probability $\mathbf{Pr}_{x\sim G_y}[C(x) = 1] = \mathbf{E}_{x\sim G_y}[C(x)] = \langle G_y, C\rangle$. In a certain sense, h serves as a *projection* of C to the domain of f. Write $C(x) = \prod_{i=1}^n C_i(x_i)$ where C_i is a conjunction depending only on x_i. Since G_y is a product distribution,

$$\langle G_y, C\rangle \;=\; \textstyle\prod_i \langle G_{y_i}, C_i\rangle \;=:\; \prod_i p_{i,y_i},$$

where we wrote $p_{i,v} \coloneqq \langle G_v, C_i\rangle \in \mathbb{R}_{\geq 0}$ for short. Fix $y^* \in \{0,1\}^n$ such that $p_{i,y_i^*} \geq p_{i,1-y_i^*}$ for all i. We now define $h(y)$ that computes $\langle G_y, C\rangle$:

$$h(y) \;\coloneqq\; \textstyle\prod_{i=1}^n \big(p_{i,1-y_i^*} + \underbrace{(p_{i,y_i^*} - p_{i,1-y_i^*})}_{\geq 0} \cdot \ell_i\big) \quad \text{where literal } \ell_i \text{ is } \begin{cases} y_i & \text{if } y_i^* = 1, \\ \bar{y}_i & \text{if } y_i^* = 0. \end{cases} \tag{8}$$

This product expression can be expanded into a conical combination of conjunctions, $h = \sum w_T T$, in the natural way, but the above "implicit" form is more concise.

Next, we record two properties of h that will suffice for the remaining analysis.

▶ **Lemma 5.1.** $\mathrm{adeg}_y(h; b_0, b_1) = \sum_i \langle \Phi_{y_i}, C_i\rangle_{\geq 0} \prod_{j\neq i} \langle G_{y_j}, C_j\rangle$.

Proof. Write $h = \sum w_T T$. We compute the average degree by summing together the weights $\sum_{T\ni \ell_i} w_T T(y)$ contributed by each of the n literals ℓ_i, i.e.,

$$\mathrm{adeg}_y(h; b_0, b_1) \;=\; \textstyle\sum_i |\ell_i|_{b_0,b_1} \cdot \sum_{T\ni\ell_i} w_T T(y).$$

If i is such that $y_i \neq y_i^*$, we have $\ell_i(y) = 0$ and so $T(y) = 0$ for all $T \ni \ell_i$; hence ℓ_i contributes no weight in this case. Suppose then that i is such that $y_i = y_i^*$; here we can write

$$h(y) \;=\; p_{i,1-y_i} \textstyle\prod_{j\neq i} p_{j,y_j} \;+\; \ell_i \cdot (p_{i,y_i} - p_{i,1-y_i}) \prod_{j\neq i} p_{j,y_j}.$$

The conjunctions T underlying the first term do not involve ℓ_i, so they contribute no weight for ℓ_i. The conjunctions T underlying the second term all involve ℓ_i and contribute a total weight of $(p_{i,y_i} - p_{i,1-y_i}) \prod_{j \neq i} p_{j,y_j}$. Altogether we get

$$\begin{aligned}
\operatorname{adeg}_y(h; b_0, b_1) &= \textstyle\sum_i |\ell_i|_{b_0,b_1} \cdot \sum_{T \ni \ell_i} w_T T(y) \\
&= \textstyle\sum_{i: y_i = y_i^*} b_{y_i} \cdot (p_{i,y_i} - p_{i,1-y_i}) \prod_{j \neq i} p_{j,y_j} \\
&= \textstyle\sum_i b_{y_i} (p_{i,y_i} - p_{i,1-y_i})_{\geq 0} \prod_{j \neq i} p_{j,y_j} \\
&= \textstyle\sum_i b_{y_i} (\langle G_{y_i}, C_i \rangle - \langle G_{1-y_i}, C_i \rangle)_{\geq 0} \prod_{j \neq i} \langle G_{y_j}, C_j \rangle \\
&= \textstyle\sum_i \langle b_{y_i} (G_{y_i} - G_{1-y_i}), C_i \rangle_{\geq 0} \prod_{j \neq i} \langle G_{y_j}, C_j \rangle \\
&= \textstyle\sum_i \langle \Phi_{y_i}, C_i \rangle_{\geq 0} \prod_{j \neq i} \langle G_{y_j}, C_j \rangle. \qquad \blacktriangleleft
\end{aligned}$$

▶ **Lemma 5.2.** *$\langle \Gamma, h \rangle = \langle \Gamma^G, C \rangle$ for all $\Gamma \colon \{0,1\}^n \to \mathbb{R}$.*

Proof. We calculate

$$\begin{aligned}
\langle \Gamma, h \rangle &= \textstyle\sum_y \Gamma(y) h(y) = \sum_y \Gamma(y) \langle G_y, C \rangle = \sum_y \Gamma(y) \big[\sum_x G_y(x) C(x) \big] \\
&= \textstyle\sum_x \big[\sum_y \Gamma(y) G_y(x) \big] C(x) = \sum_x \Gamma^G(x) C(x) = \langle \Gamma^G, C \rangle. \qquad \blacktriangleleft
\end{aligned}$$

Analysis. Let us expand the right hand side of the desired inequality (7):

$$\begin{aligned}
\operatorname{adeg}_{D_v}(C) &= |C| \cdot \langle F_v^G, C \rangle \\
&= \mathbf{E}_{y \sim F_v} \big[|C| \cdot \langle G_y, C \rangle \big] \\
&= \mathbf{E}_{y \sim F_v} \big[\big(\textstyle\sum_i |C_i| \big) \cdot \prod_i \langle G_{y_i}, C_i \rangle \big] \\
&= \mathbf{E}_{y \sim F_v} \big[\textstyle\sum_i |C_i| \langle G_{y_i}, C_i \rangle \prod_{j \neq i} \langle G_{y_j}, C_j \rangle \big] \\
&= \mathbf{E}_{y \sim F_v} \big[\textstyle\sum_i \operatorname{adeg}_{G_{y_i}}(C_i) \prod_{j \neq i} \langle G_{y_j}, C_j \rangle \big].
\end{aligned}$$

Substituting our hypothesis $\operatorname{adeg}_{G_{y_i}}(C_i) \geq \langle \Phi_{y_i}, C_i \rangle_{\geq 0} + \langle \hat{\Phi}_{y_i}, C_i \rangle$ into the above, we obtain

$$\operatorname{adeg}_{D_v}(C) \geq \underbrace{\mathop{\mathbf{E}}_{y \sim F_v} \Big[\sum_i \langle \Phi_{y_i}, C_i \rangle_{\geq 0} \prod_{j \neq i} \langle G_{y_j}, C_j \rangle \Big]}_{\textbf{(I)}} + \underbrace{\mathop{\mathbf{E}}_{y \sim F_v} \Big[\sum_i \langle \hat{\Phi}_{y_i}, C_i \rangle \prod_{j \neq i} \langle G_{y_j}, C_j \rangle \Big]}_{\textbf{(II)}}.$$

For the first term,

$$\begin{aligned}
\textbf{(I)} &= \mathbf{E}_{y \sim F_v} \big[\operatorname{adeg}_y(h; b_0, b_1) \big] && \text{(Lemma 5.1)} \\
&= \operatorname{adeg}_{F_v}(h; b_0, b_1) \\
&\geq \langle \Psi_v, h \rangle_{\geq 0} + \langle \hat{\Psi}_v, h \rangle && \text{(Feasibility of } \{\Psi_v, \hat{\Psi}_v\} \text{ and (2))} \\
&= \langle \Psi_v^G, C \rangle_{\geq 0} + \langle \hat{\Psi}_v^G, C \rangle. && \text{(Lemma 5.2)}
\end{aligned}$$

For the second term,

$$\begin{aligned}
\textbf{(II)} &= \mathbf{E}_{y \sim F_v} \big[\textstyle\sum_i \langle (G^{\leftarrow i} \hat{\Phi})_y, C \rangle \big] \\
&= \big\langle \textstyle\sum_i F_v^{G^{\leftarrow i} \hat{\Phi}}, C \big\rangle.
\end{aligned}$$

Combining these yields (7). This concludes the proof of Theorem 4.2.

6 Approximate Degree Lower Bounds

In this section we prove Theorems 2.1–2.2 using the Composition Theorem. We begin by observing that $(a_0, a_1; b_0, b_1)$-certificates $\{\Psi_v, \hat{\Psi}_v\}_{v=0,1}$ also yield lower bounds for approximate degree, if the 1-norm $\|\hat{\Psi}_1\|_1$ is not too large. We call $\{\Psi_v, \hat{\Psi}_v\}_{v=0,1}$ an $(a_0, a_1; b_0, b_1; c)$-*certificate* if $\max_v \|\hat{\Psi}_v\|_1 \leq c$.

▶ **Lemma 6.1.** *Suppose f admits an $(a_0, a_1; 1, 1; c)$-certificate. If $\epsilon \leq 1/4$ and $c \cdot \epsilon \leq a_1/4$, then $\deg_\epsilon(f) \geq \Omega(a_1)$.*

Proof. Fix a certificate $\{\Psi_v, \hat{\Psi}_v\}_{v=0,1}$ for f and suppose $\deg_\epsilon(f) = \deg(h)$ where h is a conical junta with $\|h - f\|_\infty \leq \epsilon$. Since $h(x) \leq 1 + \epsilon$ for all x, we have $\deg(h) \geq (1+\epsilon)^{-1} \operatorname{adeg}(h) \geq \Omega(\operatorname{adeg}(h))$. Now we calculate

$$\begin{aligned}
\operatorname{adeg}(h) &\geq \langle \Psi_1 + \hat{\Psi}_1, h \rangle && \text{(as in (2))} \\
&= \langle \Psi_1 + \hat{\Psi}_1, f \rangle + \langle \Psi_1 + \hat{\Psi}_1, h - f \rangle \\
&\geq a_1 - |\langle \Psi_1 + \hat{\Psi}_1, h - f \rangle| \\
&\geq a_1 - \|\Psi_1 + \hat{\Psi}_1\|_1 \cdot \|h - f\|_\infty \\
&\geq a_1 - (\|\Psi_1\|_1 + \|\hat{\Psi}_1\|_1) \cdot \epsilon \\
&\geq a_1 - (2a_1 + c) \cdot \epsilon \\
&\geq a_1/4. && \blacktriangleleft
\end{aligned}$$

We use the following version of the Composition Theorem where the bounds on 1-norms (following immediately from the definition (6)) are made explicit.

▶ **Theorem 6.2.** *Suppose f admits an $(a_0, a_1; b_0, b_1; c)$-certificate and g admits a $(b_0, b_1; 1, 1; d)$-certificate. Then $f \circ g^n$ admits an $(a_0, a_1; 1, 1; c + nd)$-certificate.*

6.1 Proof of Theorem 2.1

We iteratively apply Theorem 6.2 as follows.

1. Assume we have an $(\alpha_k, \beta_k; 1, 1; \gamma_k)$-certificate for $\mathsf{NAND}^{\circ k}$ where $\gamma_k \geq \alpha_k, \beta_k$.
2. Obtain a $(2\beta_k, \alpha_k + \frac{1}{2}\beta_k; \alpha_k, \beta_k; \beta_k)$-certificate for NAND from Figure 2.
3. Compose the above to get an $(\alpha_{k+1}, \beta_{k+1}; 1, 1; \gamma_{k+1})$-certificate for $\mathsf{NAND}^{\circ(k+1)}$ where

$$\begin{aligned}
\alpha_{k+1} &\coloneqq 2\beta_k, \\
\beta_{k+1} &\coloneqq \alpha_k + \beta_k/2, \\
\gamma_{k+1} &\coloneqq \beta_k + 2\gamma_k.
\end{aligned}$$

Note that $\alpha_{k+1}, \beta_{k+1} \leq \gamma_{k+1} \leq 3\gamma_k$. Starting with $\alpha_0 = \beta_0 = \gamma_0 = 1$ these recurrences (famously [28]) evaluate to $\alpha_k, \beta_k = \Theta(n^{0.753\cdots})$ where $n \coloneqq 2^k$. In addition, $\gamma_k \leq 3^k \leq n^{1.6}$. Now take $\epsilon \leq 1/n$ in Lemma 6.1 to prove Theorem 2.1.

6.2 Computer search for certificates

Iteratively composing (scaled versions of) the $(5/2, 5/2; 1, 1)$-certificate given in Figure 3 would yield only an $\Omega(2.5^k)$ lower bound for $\mathsf{Maj}_3^{\circ k}$. This is the best possible for our approach if we were to just compose certificates for individual Maj_3 functions. To obtain a better lower

Table 1 Certificates for $\mathsf{Maj}_3^{\circ\ell}$ for heights $\ell = 1, 2, 3$. The table lists $(\alpha_\ell, \alpha_\ell; 1, 1)$-certificates with values $\alpha_1 = 5/2$ (also illustrated in Figure 3), $\alpha_2 = 20/3$, and $\alpha_3 = 35/2$. Only $\Psi_1, \hat{\Psi}_1$ are shown as $\Psi_0, \hat{\Psi}_0$ are defined dually. We give the total weight for each equivalence class of inputs; the functions are uniform on each class. For height $\ell = 3$ we represent the inputs to the bottom-most Maj_3 gates by their Hamming weight, e.g., $001 \rightsquigarrow 1$, $011 \rightsquigarrow 2$, etc.

Function	Class representative	Class size	Ψ_1	$\hat{\Psi}_1$	$\Psi_1 + \hat{\Psi}_1$
$\mathsf{Maj}_3^{\circ 1}$	$(0,0,1)$	3	$-5/2$	0	$-5/2$
	$(0,1,1)$	3	$5/2$	$1/2$	3
	$(1,1,1)$	1	0	$-1/2$	$-1/2$
	All others		0	0	0
$\mathsf{Maj}_3^{\circ 2}$	$(001,001,011)$	81	$-20/3$	0	$-20/3$
	$(001,011,011)$	81	$20/3$	$7/3$	9
	$(000,011,011)$	27	0	$-1/3$	$-1/3$
	$(001,011,111)$	54	0	$-2/3$	$-2/3$
	$(011,011,011)$	27	0	$-4/3$	$-4/3$
	All others		0	0	0
$\mathsf{Maj}_3^{\circ 3}$	$(112,112,122)$	1594323	$-35/2$	0	-35/2
	$(112,122,122)$	1594323	$35/2$	$19/2$	27
	$(122,122,122)$	531441	0	$-7/2$	$-7/2$
	$(112,122,222)$	1062882	0	-2	-2
	$(112,122,123)$	2125764	0	$-4/3$	$-4/3$
	$(112,122,022)$	1062882	0	$-2/3$	$-2/3$
	$(111,122,122)$	531441	0	$-5/6$	$-5/6$
	$(113,122,122)$	531441	0	$-1/2$	$-1/2$
	$(012,122,122)$	1062882	0	$-2/3$	$-2/3$
	All others		0	0	0

bound, we can instead directly find a certificate for $\mathsf{Maj}_3^{\circ\ell}$ where ℓ is a small constant, and then compose that certificate. Table 1 gives certificates for $\mathsf{Maj}_3^{\circ\ell}$ for height up to $\ell = 3$. We used a computer to solve the dual LP (Dual), with the additional restriction that Ψ $(= \Psi_1 + \hat{\Psi}_1)$ should be balanced. The best balanced Ψ happened to satisfy the other conditions required by our Definition 4.1.

Notes on implementation. For computational efficiency, it is useful to prune the search space by eliminating symmetries. The symmetries of $\mathsf{Maj}_3^{\circ\ell}$ (permutations of input coordinates that do not change the value of the function) are the symmetries of the underlying height-ℓ ternary tree. These symmetries partition the set of inputs and the set of conjunctions into *equivalence classes*: two inputs/conjunctions are "equivalent" if one can be mapped to the other by a symmetry. The set of feasible solutions to the LP is also invariant under these symmetries. It follows that we may look w.l.o.g. for a Ψ that is invariant, i.e., uniform on each equivalence class. (Indeed, if Ψ is any feasible solution, we obtain an invariant solution by averaging Ψ over all the symmetries.) Thus we need only maintain one variable in the LP per equivalence class $\mathcal{X} \subseteq \{0,1\}^n$ recording the *total weight* $\sum_{x \in \mathcal{X}} \Psi(x)$ of that class. Also, for such invariant Ψ, we need only check the LP feasibility constraint $\langle \Psi, C \rangle \leq \mathrm{adeg}_{D_1}(C; b_0, b_1)$ for a single representative C from each class of conjunctions.

The optimal height-2 certificate happens to have the same *support* as the certificate produced by our Composition Theorem starting with two height-1 certificates. Inspired

by this, in order to speed up the search for height 3, we only optimised over those Ψ whose support coincides with that coming from the Composition Theorem – this LP has only 9 variables (i.e., equivalence classes of inputs), but well over 100,000 constraints (i.e., equivalence classes of conjunctions).

It is open to analyse height 4. Is there an efficient separation oracle for (Dual)?

6.3 Proof of Theorem 2.2

Table 1 defines a certificate for $\mathsf{Maj}_3^{\circ 3}$ with parameters $(35/2, 35/2; 1, 1; 19)$ and we may scale the certificate by any scalar $\alpha \geq 0$ to obtain one with parameters $((35/2)\alpha, (35/2)\alpha; \alpha, \alpha; 19\alpha)$. Using Theorem 6.2 iteratively as in Section 6.1, we get a certificate for $\mathsf{Maj}_3^{\circ k}$ with parameters

$$((35/2)^{k/3}, (35/2)^{k/3}; 1, 1; 28^{k/3} \cdot 19).$$

Here $(35/2)^{k/3} \geq n^{0.8}$ and $28^{k/3} \cdot 19 \leq n^{1.1}$ where $n := 3^k$. Hence we may apply Lemma 6.1 with $\epsilon \leq 1/n$ to conclude an ϵ-approximate degree lower bound of $\Omega((35/2)^{k/3}) = \Omega(2.59\ldots^k)$.

7 Communication Lower Bounds

In this section we prove Theorems 2.3–2.4 by applying the main result of [13]: a simulation of randomised communication protocols by conical juntas. To this end, let $\mathsf{IP}_b \colon \{0,1\}^b \times \{0,1\}^b \to \{0,1\}$ be the two-party (Alice has x, Bob has y) inner-product function given by

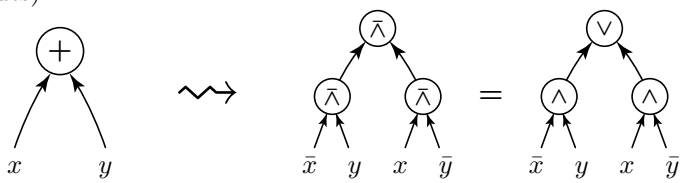
$$\mathsf{IP}_b(x, y) \;:=\; \langle x, y \rangle \bmod 2.$$

Let $\mathsf{BPP}^{\mathsf{cc}}_\epsilon(F)$ denote the randomised ϵ-error communication complexity of $F \colon \mathcal{X} \times \mathcal{Y} \to \{0,1\}$. The following is a corollary of [13, Theorem 31] (the original formulation there talks about $\mathsf{WAPP}^{\mathsf{dt}}_\epsilon(f)$ which is the same as $\deg_\epsilon(f)$; moreover, the result is stated for $\epsilon = \Theta(1)$, but the theorem is true more generally for $\epsilon = 2^{-\Theta(b)}$).

▶ **Theorem 7.1** ([13]). *Let $\epsilon := 1/n$ and $b := \Theta(\log n)$ (with a large enough implicit constant). For any $f \colon \{0,1\}^n \to \{0,1\}$ we have*

$$\mathsf{BPP}^{\mathsf{cc}}_{\epsilon/2}(f \circ \mathsf{IP}_b^n) \;\geq\; \Omega(\deg_\epsilon(f) \cdot b).$$

Let us prove Theorem 2.3 (a similar argument works for Theorem 2.4). A key observation (also made in [16, §3]) is that $\mathsf{IP}_b = \mathsf{XOR}_b \circ \mathsf{AND}^b$ reduces to computing a binary NAND tree on $O(b^2)$ bits. To see this, think of the b-bit parity function XOR_b as a height-$(\log b)$ binary tree of XOR gates. Each such XOR gate can be rewritten as a height-2 NAND tree (with some negations on inputs):



In the binary XOR tree, replace the top XOR gate with this NAND tree (this involves making copies of some subtrees), push the negations to inputs, and repeat recursively. This gives us a height-$(2 \log b)$ NAND tree. Moreover, the bottom layer of AND gates in IP_b is also easily simulated by NAND gates. Consequently, for some $N := \Theta(nb^2)$, the communication matrix of $\mathsf{NAND}^{\circ \log n} \circ \mathsf{IP}_b^n$ appears as a submatrix of $\mathsf{NAND}^{\circ \log N}$ (relative to some bipartition of the input given by the reduction).

We can now derive Theorem 2.3 – here ϵ and b are defined as in Theorem 7.1, and $\gtrsim$ means that we ignore polylog(N) factors.

$$\begin{aligned}
\mathsf{BPP}^{\mathsf{cc}}_{1/3}(\mathsf{NAND}^{\circ \log N}) &\gtrsim \mathsf{BPP}^{\mathsf{cc}}_{\epsilon/2}(\mathsf{NAND}^{\circ \log N}) && \text{(Error reduction)}\\
&\gtrsim \mathsf{BPP}^{\mathsf{cc}}_{\epsilon/2}(\mathsf{NAND}^{\circ \log n} \circ \mathsf{IP}^n_b) && \text{(Key observation)}\\
&\gtrsim \deg_\epsilon(\mathsf{NAND}^{\circ \log n}) && \text{(Theorem 7.1)}\\
&\gtrsim n^{0.753\ldots} && \text{(Theorem 2.1)}\\
&= \tilde{\Theta}(N^{0.753\ldots}).
\end{aligned}$$

Acknowledgements. Many thanks to Thomas Watson for a careful perusal of the manuscript, to Li-Yang Tan for correspondence, and to anonymous referees for comments.

References

1 Kazuyuki Amano. Bounding the randomized decision tree complexity of read-once boolean functions. In *Proceedings of the 22nd Symposium on Discrete Algorithms (SODA)*, pages 1729–1744. SIAM, 2011.

2 Kazuyuki Amano. On directional vs. general randomized decision tree complexity for read-once formulas. *Chicago Journal of Theoretical Computer Science*, 2011(3), 2011. doi:10.4086/cjtcs.2011.003.

3 Kazuyuki Amano. Researching the complexity of boolean functions with computers. *Bulletin of EATCS*, 101:64–91, 2014. URL: http://bulletin.eatcs.org/index.php/beatcs/article/view/181.

4 Andris Ambainis, Kaspars Balodis, Aleksandrs Belovs, Troy Lee, Miklos Santha, and Juris Smotrovs. Separations in query complexity based on pointer functions. Technical Report TR15-098, Electronic Colloquium on Computational Complexity (ECCC), 2015. URL: http://eccc.hpi-web.de/report/2015/098/.

5 Andris Ambainis, Andrew Childs, Ben Reichardt, Robert Špalek, and Shengyu Zhang. Any AND-OR formula of size N can be evaluated in time $N^{1/2+o(1)}$ on a quantum computer. *SIAM Journal on Computing*, 39(6):2513–2530, 2010. doi:10.1137/080712167.

6 Paul Beame and Joan Lawry. Randomized versus nondeterministic communication complexity. In *Proceedings of the 24th Symposium on Theory of Computing (STOC)*, pages 188–199. ACM, 1992. doi:10.1145/129712.129732.

7 Aleksandrs Belovs. Non-intersecting complexity. In *Proceedings of the 32nd Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM)*, pages 158–165. Springer, 2006. doi:10.1007/11611257_13.

8 Harry Buhrman and Ronald de Wolf. Complexity measures and decision tree complexity: A survey. *Theoretical Computer Science*, 288(1):21–43, 2002. doi:10.1016/S0304-3975(01)00144-X.

9 Mark Bun and Justin Thaler. Dual lower bounds for approximate degree and Markov–Bernstein inequalities. *Information and Computation*, 243:2–25, 2015. doi:10.1016/j.ic.2014.12.003.

10 Siu On Chan, James Lee, Prasad Raghavendra, and David Steurer. Approximate constraint satisfaction requires large LP relaxations. In *Proceedings of the 54th Symposium on Foundations of Computer Science (FOCS)*, pages 350–359. IEEE, 2013. doi:10.1109/FOCS.2013.45.

11 Ehud Friedgut, Jeff Kahn, and Avi Wigderson. Computing graph properties by randomized subcube partitions. In *Proceedings of the 6th International Workshop on Randomization and Computation (RANDOM)*, pages 105–113. Springer, 2002. doi:10.1007/3-540-45726-7_9.

12 Mika Göös. Lower bounds for clique vs. independent set. In *Proceedings of the 56th Symposium on Foundations of Computer Science (FOCS)*, pages 1066–1076. IEEE, 2015. doi:10.1109/FOCS.2015.69.

13 Mika Göös, Shachar Lovett, Raghu Meka, Thomas Watson, and David Zuckerman. Rectangles are nonnegative juntas. In *Proceedings of the 47th Symposium on Theory of Computing (STOC)*, pages 257–266. ACM, 2015. (Full version: http://eccc.hpi-web.de/report/2014/147/). doi:10.1145/2746539.2746596.

14 Mika Göös, Toniann Pitassi, and Thomas Watson. Deterministic communication vs. partition number. In *Proceedings of the 56th Symposium on Foundations of Computer Science (FOCS)*, pages 1077–1088. IEEE, 2015. doi:10.1109/FOCS.2015.70.

15 Rahul Jain and Hartmut Klauck. The partition bound for classical communication complexity and query complexity. In *Proceedings of the 25th Conference on Computational Complexity (CCC)*, pages 247–258. IEEE, 2010. doi:10.1109/CCC.2010.31.

16 Rahul Jain, Hartmut Klauck, and Shengyu Zhang. Depth-independent lower bounds on the communication complexity of read-once boolean formulas. In *Proceedings of the 16th International Computing and Combinatorics Conference (COCOON)*, pages 54–59. Springer, 2010. doi:10.1007/978-3-642-14031-0_8.

17 Rahul Jain, Troy Lee, and Nisheeth Vishnoi. A quadratically tight partition bound for classical communication complexity and query complexity. Technical report, arXiv, 2014. arXiv:1401.4512.

18 T.S. Jayram, Swastik Kopparty, and Prasad Raghavendra. On the communication complexity of read-once AC^0 formulae. In *Proceedings of the 24th Conference on Computational Complexity (CCC)*, pages 329–340, 2009. doi:10.1109/CCC.2009.39.

19 T.S. Jayram, Ravi Kumar, and D. Sivakumar. Two applications of information complexity. In *Proceedings of the 35th Symposium on Theory of Computing (STOC)*, pages 673–682. ACM, 2003. doi:10.1145/780542.780640.

20 Jedrzej Kaniewski, Troy Lee, and Ronald de Wolf. Query complexity in expectation. In *Proceedings of the 42nd International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 761–772. Springer, 2015. doi:10.1007/978-3-662-47672-7_62.

21 Robin Kothari, David Racicot-Desloges, and Miklos Santha. Separating decision tree complexity from subcube partition complexity. In *Proceedings of the 19th International Workshop on Randomization and Computation (RANDOM)*, Leibniz International Proceedings in Informatics (LIPIcs), pages 915–930. Schloss Dagstuhl, 2015. doi:10.4230/LIPIcs.APPROX-RANDOM.2015.915.

22 Itamar Landau, Asaf Nachmias, Yuval Peres, and Sithparran Vanniasegaram. The lower bound for evaluating a recursive ternary majority function: an entropy-free proof. Technical report, U.C. Berkeley, 2006.

23 Joan Lawry. *Communication complexity: Iterative techniques for lower bounds.* PhD thesis, University of Washington, 1993. UW-CSE-93–3-04.

24 James Lee, Prasad Raghavendra, and David Steurer. Lower bounds on the size of semidefinite programming relaxations. In *Proceedings of the 47th Symposium on Theory of Computing (STOC)*, pages 567–576. ACM, 2015. doi:10.1145/2746539.2746599.

25 Nikos Leonardos. An improved lower bound for the randomized decision tree complexity of recursive majority. In *Proceedings of the 40th International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 696–708. Springer, 2013. doi:10.1007/978-3-642-39206-1_59.

26 Nikos Leonardos and Michael Saks. Lower bounds on the randomized communication complexity of read-once functions. *Computational Complexity*, 19(2):153–181, 2010. doi:10.1007/s00037-010-0292-2.

27 Frédéric Magniez, Ashwin Nayak, Miklos Santha, Jonah Sherman, Gábor Tardos, and David Xiao. Improved bounds for the randomized decision tree complexity of recursive majority. *Random Structures & Algorithms*, 2015. In press. doi:10.1002/rsa.20598.

28 Michael Saks and Avi Wigderson. Probabilistic boolean decision trees and the complexity of evaluating game trees. In *Proceedings of the 27th Symposium on Foundations of Computer Science (FOCS)*, pages 29–38. IEEE, 1986. doi:10.1109/SFCS.1986.44.

29 Miklos Santha. On the Monte Carlo boolean decision tree complexity of read-once formulae. *Random Structures & Algorithms*, 6(1):75–87, 1995. doi:10.1002/rsa.3240060108.

30 Petr Savický. On determinism versus unambiguous nondeterminism for decision trees. Technical Report TR02-009, Electronic Colloquium on Computational Complexity (ECCC), 2002. URL: http://eccc.hpi-web.de/report/2002/009/.

31 Alexander Sherstov. Approximating the AND-OR tree. *Theory of Computing*, 9(20):653–663, 2013. doi:10.4086/toc.2013.v009a020.

32 Alexander Sherstov. Breaking the Minsky-Papert barrier for constant-depth circuits. In *Proceedings of the 46th Symposium on Theory of Computing (STOC)*, pages 223–232. ACM, 2014. doi:10.1145/2591796.2591871.

33 Marc Snir. Lower bounds on probabilistic linear decision trees. *Theoretical Computer Science*, 38:69–82, 1985. doi:10.1016/0304-3975(85)90210-5.

34 Nikolai Vereshchagin. Randomized boolean decision trees: Several remarks. *Theoretical Computer Science*, 207(2):329–342, 1998. doi:10.1016/S0304-3975(98)00071-1.

Tight Bounds for Communication-Assisted Agreement Distillation*

Venkatesan Guruswami[1] and Jaikumar Radhakrishnan[2]

1 **Computer Science Department, Carnegie Mellon University, Pittsburgh, USA**
guruswami@cmu.edu

2 **School of Technology and Computer Science, Tata Institute of Fundamental Research, Mumbai, India**
jaikumar@tifr.res.in

Abstract

Suppose Alice holds a uniformly random string $X \in \{0,1\}^N$ and Bob holds a noisy version Y of X where each bit of X is flipped independently with probability $\varepsilon \in [0, \frac{1}{2}]$. Alice and Bob would like to extract a common random string of min-entropy at least k. In this work, we establish the communication versus success probability trade-off for this problem by giving a protocol and a matching lower bound (under the restriction that the string to be agreed upon is determined by Alice's input X). Specifically, we prove that in order for Alice and Bob to agree on a common string with probability $2^{-\gamma k}$ ($\gamma k \geqslant 1$), the optimal communication (up to $o(k)$ terms, and achievable for large N) is precisely $(C(1-\gamma) - 2\sqrt{C(1-C)\gamma})k$, where $C := 4\varepsilon(1-\varepsilon)$. In particular, the optimal communication to achieve $\Omega(1)$ agreement probability approaches $4\varepsilon(1-\varepsilon)k$.

We also consider the case when Y is the output of the binary erasure channel on X, where each bit of Y equals the corresponding bit of X with probability $1-\varepsilon$ and is otherwise erased (that is, replaced by a '?'). In this case, the communication required becomes $(\varepsilon(1-\gamma) - 2\sqrt{\varepsilon(1-\varepsilon)\gamma})k$. In particular, the optimal communication to achieve $\Omega(1)$ agreement probability approaches εk, and with no communication the optimal agreement probability approaches $2^{-\frac{1-\sqrt{1-\varepsilon}}{1+\sqrt{1-\varepsilon}}k}$.

Our protocols are based on covering codes and extend the approach of (Bogdanov and Mossel, 2011) for the zero-communication case. Our lower bounds rely on hypercontractive inequalities. For the model of bit-flips, our argument extends the approach of (Bogdanov and Mossel, 2011) by allowing communication; for the erasure model, to the best of our knowledge the needed hypercontractivity statement was not studied before, and it was established (given our application) by (Nair and Wang 2015). We also obtain information complexity lower bounds for these tasks, and together with our protocol, they shed light on the recently popular "most informative Boolean function" conjecture of Courtade and Kumar.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases communication complexity, covering codes, hypercontractivity, information theory, lower bounds, pseudorandomness

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.6

* Research supported in part by United States National Science Foundation grants CCF-0963975 and CCF-1422045, and by the hospitality of the Simons Institute for the Theory of Computing, Berkeley.

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 6; pp. 6:1–6:17

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

1 Introduction

Suppose Alice holds a string $X = (x_1, x_2, \dots)$ of uniformly random bits and Bob holds a correlated random string $Y = (y_1, y_2, \dots)$ where the bit y_j is the bit x_j flipped (independently for each j) with probability $\varepsilon \in (0, 1)$. Their goal is to communicate as little as possible and agree on a uniformly random string in $\{0,1\}^k$ (or under a relaxed requirement, sample a common string from a distribution of min-entropy at least k).

Besides being a natural task, this scenario also relates to the problem of extracting a unique ID from process variations; see [5], which studied the communication free version of this question, for further discussion of this motivation. The agreement distillation problem also naturally arises in the context of simulating communication protocols that use perfect shared randomness when the parties only share correlated randomness, and was recently studied with this motivation in [6]. The underlying information-theoretic question, on the maximum information a function of X can convey about its noisy version Y, has also received widespread interest lately, following the appealing conjecture made in [9] that a dictator (or canalizing) function is the most informative Boolean function (the one maximizing $I[f(X) : Y]$).

Our work is a follow-up to [6, 5] and is motivated by questions such as: How many bits of communication are needed for the agreement distillation task to succeed with high probability? At the other extreme, what is the best success probability of a strategy that involves no communication? More precisely, what is the trade-off between communication and success probability?

Note that there are two trivial protocols: one where Alice simply sends the first k bits of X to Bob (which achieves agreement probability of 1), and a zero-communication protocol where both players simply output their first k bits as the common randomness (which achieves agreement probability of $(1-\varepsilon)^k$). The former protocol does not exploit the fact that Bob holds a string Y which is correlated with X. How much can we leverage this to save on communication while at the same time ensuring good agreement probability? A simple protocol based on capacity-achieving codes for the binary symmetric channel was given in [6] with communication $(h(\varepsilon) + o(1))k$ and high agreement probability; in [6], an $\Omega(\varepsilon k)$ lower bound based on [5] was also observed. This established that a factor $c(\varepsilon)$ savings in communication is the best one can hope for, but left a gap even in the asymptotic growth of $c(\varepsilon)$.

1.1 Our results

We obtain tight communication complexity upper and lower bounds for the above problem, identifying the precise trade-off between communication and agreement probability (see Theorem 1.1 below, our bounds are sharp up to $o(k)$ bits). Our upper bounds are achieved by one-way communication protocols where Alice sends a single message to Bob. Our lower bounds hold for a slightly more general model where Alice's output depends only on her input X, but Bob's output may depended on his input Y and the transcript of an arbitrary two-way interaction with Alice. Below is a statement of the bounds we get.

▶ **Theorem 1.1.** *Let $\gamma \in [0, 1]$, $\varepsilon \in [0, 1/2]$, and $k \geqslant 1$ be an integer. Consider the above setting where Alice and Bob have uniformly random strings X and Y (of sufficiently large length compared to k) that differ in each position independently with probability ε. The goal is for Alice and Bob to agree on a shared string $g_A(X)$, which only depends on Alice's input X. Define $C := 4\varepsilon(1-\varepsilon)$.*

- (Upper bound) *There is a protocol where $g_A(X)$ is uniformly distributed in $\{0,1\}^k$, and Alice sends $(C(1-\gamma)-2\sqrt{C(1-C)\gamma})k$ bits to Bob, who then succeeds in guessing $g_A(X)$ with probability at least $2^{-\gamma k - O(\log k)}$.*
- (Matching lower bound) *Suppose there is a protocol with $H_\infty[g_A(X)] \geqslant k$ where Alice and Bob exchange c bits after which Bob is able to guess $g_A(X)$ with probability $2^{-\gamma k}$. Then*

$$c \geqslant (C(1-\gamma) - 2\sqrt{C(1-C)\gamma})k \ .$$

In particular, this implies that for large k, to achieve agreement probability $\Theta(1)$ the optimal communication approaches $4\varepsilon(1-\varepsilon)k$, and with zero communication the best achievable success probability approaches $2^{-\frac{\varepsilon}{1-\varepsilon}k}$.[1]

Note that in the above setup, Bob's input Y can be viewed as X that is distorted by a binary symmetric channel, $\mathsf{BSC}(\varepsilon)$, which flips each bit independently with probability ε. Inspired by this view, one can consider a similar problem for other discrete memoryless channels relating X and Y. We consider the binary erasure channel, $\mathsf{BEC}(\varepsilon)$, where each Y_j equals X_j with probability $1-\varepsilon$ and is erased (say, replaced by a '?') with probability ε, and obtain tight upper and lower bounds for this setting as well (basically the quantity $C = 4\varepsilon(1-\varepsilon)$ is replaced by ε in the bounds).

▶ **Theorem 1.2.** *Let $\gamma, \varepsilon \in [0,1]$ and $k \geqslant 1$ be an integer. For the agreement distillation problem when Y is obtained by passing X through $\mathsf{BEC}(\varepsilon)$, the following hold.*

- (Upper bound) *There is a protocol where $g_A(X)$ is uniformly distributed in $\{0,1\}^k$, and Alice sends $(\varepsilon(1-\gamma)-2\sqrt{\varepsilon(1-\varepsilon)\gamma})k$ bits to Bob, who succeeds in guessing $g_A(X)$ with probability at least $2^{-\gamma k - O(\log k)}$.*
- (Matching lower bound) *Suppose there is a protocol where $H[g_A(X)] \geqslant k$ and Alice and Bob exchange c bits after which Bob is able to guess $g_A(X)$ with probability $2^{-\gamma k}$. Then $c \geqslant (\varepsilon(1-\gamma) - 2\sqrt{\varepsilon(1-\varepsilon)\gamma})k$.*

In particular, it can be shown that, for large k, to achieve agreement probability $\Theta(1)$ the optimal communication approaches εk, and with zero communication the best achievable success probability approaches $2^{-\frac{(1-\sqrt{1-\varepsilon})k}{1+\sqrt{1-\varepsilon}}}$.

We also study information complexity bounds, proving the following lower bound on the information content needed in the protocol transcript.

▶ **Theorem 1.3.** *Let $g_A(X)$ take values in the set $\{0,1\}^{k'}$ such that $H[g_A(X)] \geqslant k$. Suppose $\pi(X,Y)$ is the transcript of a protocol that enables Bob to guess $g_A(X)$ with probability at least $1-\delta$, for some $\delta \in [0,1]$. Then we have*

- *$H[\pi(X,Y)] \geqslant 4\varepsilon(1-\varepsilon)k - \delta k' - h(\delta)$ when Y is the output of $\mathsf{BSC}(\varepsilon)$ on X;*
- *$H[\pi(X,Y)] \geqslant \varepsilon k - \delta k' - h(\delta)$ when Y is the output of $\mathsf{BEC}(\varepsilon)$ on X.*

(Note that some term like $-\delta k'$ in the lower bounds is unavoidable. For example, $g_A(X)$ might be $0^{k'}$ with probability $1-\delta$ and a uniformly random string in $\{0,1\}^{k'}$ with probability δ. If Bob produces $0^{k'}$ always, they agree with probability $1-\delta$.)

Since the entropy $H[\pi(X,Y)]$ lower bounds the length of the transcript, the above also implies lower bounds on the communication complexity. However, the bounds are good only when $\delta \to 0$, whereas Theorems 1.1 and 1.2 apply even when the success probability $1-\delta$ is very small, and imply communication lower bounds of $(4\varepsilon(1-\varepsilon) - o(1))k$ and $(\varepsilon - o(1))k$ for any constant success probability.

[1] For the problem with zero communication, lower and upper bounds in [5] already establish that the best probability of success is $2^{-\frac{\varepsilon}{1-\varepsilon}k}$ (see Section 1.2).

The communication upper bounds from Theorems 1.1 and 1.2 of course imply protocols with the same upper bounds on entropy. In particular, when the failure probability $\delta \to 0$, the optimal entropy of the transcript of an agreement distillation protocol approaches $4\varepsilon(1-\varepsilon)k$ for $\mathsf{BSC}(\varepsilon)$ and εk for $\mathsf{BEC}(\varepsilon)$.

1.2 Prior and related work

The variant of agreement distillation where the goal of the two parties is to extract a single bit without any interaction was studied independently a number of times; see [25] and references therein. It is known that in this case the optimal protocol is for the two parties to use the first bit. The works [16, 17] consider the problem of extracting a common random bit in the multi-party setting where m players receive noisy versions of a common random string; in this case for large m the majority function is close to being optimal in terms of maximizing the agreement probability. The problem of two parties agreeing on k random bits without any communication, when given strings X, Y correlated via $\mathsf{BSC}(\varepsilon)$, was considered by Bogdanov and Mossel [5]. They proved that no strategy can achieve agreement probability better than $2^{-k\varepsilon/(1-\varepsilon)}$ and also gave a protocol with agreement probability $O((k\varepsilon)^{-1/2} \cdot 2^{-k\varepsilon/(1-\varepsilon)})$ when $k \geqslant \Omega(1/\varepsilon)$.

All these results are for the model where no communication is allowed between Alice and Bob, and the goal is to maximize the agreement probability. Canonne et al. [6] considered the setting where Alice and Bob can communicate, and gave a simple scheme based on capacity-achieving codes for agreeing on k random bits with high probability when Alice sends a single message of $(h(\varepsilon) + o(1))k$ bits to Bob. They also noted an $\Omega(\varepsilon k)$ lower bound based on the agreement probability upper bound for zero communication protocols from [5]. Zhao and Chia [26] establish that to agree with high probability on a common random variable K with *Shannon entropy* $H[K] \geqslant k$, the communication required approaches precisely $(1 - \rho^2(X_1; Y_1))k$, where $\rho(A; B)$ is the Hirschfeld-Gebelein-Rényi (HGR) maximal correlation of the pair $(A; B)$ of random variables. The HGR correlation for $\mathsf{BSC}(\varepsilon)$ (resp. $\mathsf{BEC}(\varepsilon)$) equals $1-2\varepsilon$ (resp. $\sqrt{1-\varepsilon}$), so this implies the communication bounds of Theorems 1.1 and 1.2, albeit for the setting of ensuring high Shannon entropy and agreement probability tending to 1. The Shannon entropy of a random variable is lower bounded by its min-entropy, so a lower bound for distilling randomness with Shannon entropy k implies the same lower bound for min-entropy (our setting). But note that our lower bounds hold also for success probability bounded away from 1, for which we have to rely on hypercontractivity based arguments. Indeed, the main novelty in our results is the establishment of the precise trade-off between communication and probability of agreement.

Our work focuses only on the efficiency of shared randomness generation as a function of communication (and success probability). We allow the number of correlated samples $N \to \infty$ for any desired value of k, the number of shared random bits to be generated (indeed in our protocols as presented, N will be exponential in k and we did not try to optimize this trade-off). Prior work has also studied the efficiency of common randomness generation as a function of N [1, 26], specifically understanding the "CR capacity" $C(R)$ wherein $C(R)N$ bits of shared randomness can be generated (with high probability) using RN bits of communication, for a fixed $R > 0$ and growing N.[2]

[2] With zero communication, it is not possible to distill any common randomness with high probability, unless the joint distribution of X_1 and Y_1 is decomposable, which is captured by the HGR maximal correlation $\rho(X_1, Y_1)$ equaling 1 [13, 21, 15].

Turning to our information-theoretic results, the entropy lower bound in Theorem 1.3 for $\mathsf{BSC}(\varepsilon)$ is based on the following claim. Let $X^n, Y^n \in \{0,1\}^n$ be random strings with (X_i, Y_i) being i.i.d. and related via the channel $\mathsf{BSC}(\varepsilon)$. Then, for every function $g_A : \{0,1\}^n \to \{0,1\}^k$ we have $I[g_A(X^n) : Y^n] \leqslant (1-2\varepsilon)^2 k$. This upper bound on mutual information follows from the so-called Mrs. Gerber's Lemma [24]; such an upper bound was established using a limit argument in [7], and is attributed to Erkip [10] in [9].

The earlier mentioned conjecture from [9] on the most informative Boolean function asserts that when $k = 1$, we have $I[g_A(X^n) : Y^n] \leqslant 1 - h(\varepsilon)$. If this conjecture were true for every k, then one would have $I[g_A(X^n) : Y^n] \leqslant (1 - h(\varepsilon))k$ when the range of g_A is $\{0,1\}^k$. However, our communication protocol in Theorem 1.1 implies the existence of a function g_A for which

$$\begin{aligned} I[g_A(X^n) : Y^n] &= H[g_A(X^n)] - H[g_A(X^n) \mid Y^n] \\ &\geqslant k - (4\varepsilon(1-\varepsilon) + o(1))k \\ &= (1-2\varepsilon)^2 k - o(k) > (1 - h(\varepsilon))k \end{aligned}$$

(for $\varepsilon \in (0, 1/2)$). So for functions outputting a large number k of bits, the projection onto the first k bits is *not* the most informative function. This latter result was already established in the recent work [7], where a function $g_A(X^n)$ based on lossy data compression (under Hamming distortion) was shown to achieve $\liminf_{n\to\infty} I[g_A(X^n) : Y^n] \geqslant (1-2\varepsilon)^2 k$.

Our entropy lower bound in Theorem 1.3 for the case of $\mathsf{BEC}(\varepsilon)$ is based on the inequality $I[g_A(X^n) : Y^n] \leqslant (1-\varepsilon)k$ for an arbitrary function $g_A : \{0,1\}^n \to \{0,1\}$, which we establish using Shearer's lemma. So, for the erasure channel, outputting the first k bits indeed maximizes the information about the channel output Y^n, for every $k \geqslant 1$, and in particular the dictator is the most informative function when $k = 1$.

As an appealing conjecture bridging information theory and analysis of Boolean functions, the most informative function conjecture of Courtade and Kumar [9] has generated a lot of interest. Closely related problems were studied earlier by Erkip and Cover [11], and recent works addressing aspects of the Courtade-Kumar conjecture include [2, 4, 7, 20, 14, 23, 22].

1.3 Techniques in brief

Our communication protocols are extensions of the Bogdanov-Mossel protocol [5]. Their zero communication protocol for $\mathsf{BSC}(\varepsilon)$ was based on an "affine covering code" $C \subseteq \mathbb{F}_2^n$ of size 2^k, and both Alice and Bob rounded their inputs X^n and Y^n to the closest point in C (with some explicit rule in case of ties). The probabilistic method is used to establish the existence of an affine space of $\mathbb{F}_2^n$ of dimension k such that each output is generated with the same probability 2^{-k}, and the agreement probability is high (at least $\approx 2^{-\varepsilon k/(1-\varepsilon)}$). In our scheme, we use different functions for Alice and Bob, with Bob searching for a codeword in a larger radius. This will lead to a list of candidates on Bob's side, and he will use Alice's message to pick a unique element from the list. Picking parameters carefully leads us to the protocol with the optimal trade-off between communication and agreement probability claimed in Theorem 1.1. The protocol for the erasure case in Theorem 1.2 works similarly, with the analysis handling some technicalities by conditioning on the high probability event of Y having close to εN erasures.

Turning to our lower bounds, as mentioned above, our entropy lower bounds are based on Mrs. Gerber's lemma for $\mathsf{BSC}(\varepsilon)$ and Shearer's lemma for $\mathsf{BEC}(\varepsilon)$. Our communication lower bounds rely on hypercontractive inequalities for the random variables corresponding to $\mathsf{BSC}(\varepsilon)$ and $\mathsf{BEC}(\varepsilon)$. If (X_i, Y_i) are i.i.d. copies of a correlated random variable (X, Y),

and $f : X^n \to \mathbb{R}$, such a hypercontractive inequality upper bounds $\|\mathbb{E}[f(X)|Y]\|_q$ by the norm $\|f\|_p$ with $p < q$ (see Section 4.1 for the definition of these norms). The best possible relationship between p and q depends on amount of correlation between X and Y. For $\mathsf{BSC}(\varepsilon)$, it is a classical result in the analysis of Boolean functions that one can take $p = 1 + (1-2\varepsilon)^2(q-1)$ [19, Chap. 16]. The inequality for the erasure channel does not appear to have been studied before, and we use the bound $p = 1 + (1-\varepsilon)(q-1)$, shown to be valid for $1 \leqslant q \leqslant 3$ by Nair [18], prompted by our application.

The lower bound for zero-communication in [5] was also established using hypercontractivity. The reduction to an hypercontractive inequality was more direct in their case, as the success probability can be expressed as $\mathbb{E}_{(X,Y)}[g_A(X)g_B(Y)]$ which equals an inner product $\mathbb{E}_X[g_A(X)T_{1-2\varepsilon}g_B(X)]$ for the Bonami-Beckner noise operator $T_{1-2\varepsilon}$. When Alice is allowed to send a message to Bob, we need a bit more care in applying the hypercontractive inequality to deduce the lower bound. Also, as mentioned earlier, for the case of erasures, the requisite hypercontractive inequality seems to not have been studied before.

It is natural to wonder what the situation is for more general channels besides the BSC and the BEC. The lower bound on communication to achieve constant agreement probability, which approaches $4\varepsilon(1-\varepsilon)k$ and εk respectively for $\mathsf{BSC}(\varepsilon)$ and $\mathsf{BEC}(\varepsilon)$, arises from the limiting ratio $\frac{p-1}{q-1}$ as $q \downarrow 1$. For an arbitrary discrete channel $(X,Y) \sim p(x,y)$, this limit has been shown to equal

$$s^*(Y;X) := \sup_{r(y)\neq p(y)} \frac{D(r(x)||p(x))}{D(r(y)||p(y))} \tag{1}$$

where $r(x)$ denotes the x-marginal distribution of $r(x,y) = r(y)p(x|y)$ [3]. Our methods imply a communication lower bound of $(1 - s^*(Y;X))k - o(k)$ for an arbitrary channel, though we do not know if this is tight in general.

2 The model

Alice receives a random string $X = (X_1, X_2, \ldots, X_N)$ and Bob receives a (correlated) string $Y = (Y_1, Y_2, \ldots, Y_N)$. We will assume the length N of these strings is sufficiently large, but it will otherwise not play an important role (and will be mostly suppressed) in our arguments. Alice uses her random input string X to produce an output in $\{0,1\}^{k'}$. Then, based on the inputs, Alice and Bob interact using a two-party protocol σ to produce a transcript $\sigma(X,Y)$. Finally, Bob produces an output in $\{0,1\}^{k'}$ based on his input Y and $\sigma(X,Y)$. Their goal is to ensure that the outputs agree and have high min-entropy.

▶ **Definition 2.1.** A (k', k, η, R)-agreement distillation protocol for a pair of random variables $R = (X,Y)$ is a triple (g_A, g_B, σ), where σ is a two-party protocol and $g_A(X), g_B(Y, \sigma(X,Y)) \in \{0,1\}^{k'}$, such that

1. $H_\infty[g_A(X)] \geqslant k$;
2. $\Pr[g_A(X) = g_B(Y, \sigma(X,Y))] \geqslant \eta$.

Let $\Pi(k', k, \eta, R)$ be the collection of all (k', k, η, R)-protocols. For $\pi \in \Pi(k', k, \eta, R)$, let $\pi(X,Y)$ denote the transcript of the underlying two-party protocol on input (X,Y). Let

$$h^R(k', k, \eta) = \min_{\pi \in \Pi(k', k, \eta, R)} H[\pi(X,Y)]; \tag{2}$$

$$c^R(k', k, \eta) = \min_{\pi \in \Pi(k', k, \eta, R)} \max_{x,y} |\pi(x,y)|. \tag{3}$$

We will consider two joint distributions of $R = (X,Y)$ in this work, where (X_i, Y_i) are independently generated as follows.

- Binary symmetric channel, $\mathsf{BSC}(N,\varepsilon)$: X_i is uniform in $\{0,1\}$, and $Y_i = X_i$ with probability $(1-\varepsilon)$ and $Y_i = 1 - X_i$ with probability ε.
- Binary erasure channel, $\mathsf{BEC}(N,\varepsilon)$: X_i is uniform in $\{0,1\}$, and $Y_i = X_i$ with probability $(1-\varepsilon)$ and $Y_i =?$ with probability ε.

3 The entropy bounds

In this section, we show the following, which implies the lower bounds claimed in Theorem 1.3.

▶ **Theorem 3.1.** *We have the following lower bounds:*

$$h^{\mathsf{BSC}(N,\varepsilon)}(k',k,\eta) \geqslant 4\varepsilon(1-\varepsilon)k - (1-\eta)k' - h(\eta);$$
$$h^{\mathsf{BEC}(N,\varepsilon)}(k',k,\eta) \geqslant \varepsilon k - (1-\eta)k' - h(\eta).$$

Both parts of the theorem will be justified using the following idea. The channel limits the mutual information between Alice's output and Bob's input. Alice's message must, therefore, make up for the shortfall.

▶ **Claim 3.2.**

(a) *If* $(X,Y) \sim \mathsf{BSC}(N,\varepsilon)$, *then*

$$I[g_A(X) : Y] \leqslant (1-2\varepsilon)^2\, I[g_A(X) : X] = (1-2\varepsilon)^2\, H[g_A(X)]. \tag{4}$$

(b) *If* $(X,Y) \sim \mathsf{BEC}(N,\varepsilon)$, *then*

$$I[g_A(X) : Y] \leqslant (1-\varepsilon)\, I[g_A(X) : X] = (1-\varepsilon)\, H[g_A(X)]. \tag{5}$$

Proof of Theorem 3.1. First, we have

$$\begin{aligned}
\mathbb{E}[|\Pi(X,Y)|] &\geqslant H[\Pi(X,Y)] \\
&\geqslant I[\Pi(X,Y) : g_A(X)Y] \\
&= I[g_A(X) : \Pi(X,Y)Y] - I[Y : g_A(X)] + I[Y : \Pi(X,Y)] \\
&\geqslant H[g_A(X)] - H[g_A(X) \mid \Pi(X,Y)Y] - I[Y : g_A(X)] \qquad (6)\\
&\geqslant H[g_A(X)] - h(\eta) - (1-\eta)k' - I[Y : g_A(X)]. \qquad (7)
\end{aligned}$$

Our assumption implies that $H[g_A(X)] \geqslant k$. We use the claim above to bound the last term on the right.

Binary symmetric channel. From (7) and Claim 3.2 (a), we obtain

$$\begin{aligned}
\mathbb{E}[|\Pi(X,Y)] &\geqslant (1-(1-2\varepsilon)^2)H[g_A(X)] - h(\eta) - (1-\eta)k' \\
&\geqslant 4\varepsilon(1-\varepsilon)k - (1-\eta)k' - h(\eta).
\end{aligned}$$

Erasure channel. From (7) and Claim 3.2 (b), we obtain

$$\mathbb{E}[|\Pi(X,Y)] \geqslant (1-(1-\varepsilon))H[g_A(X)] - h(\eta) - (1-\eta)k' \geqslant \varepsilon k - (1-\eta)k' - h(\eta)\ .$$

◀

Proof of Claim 3.2.

(a) Recall the following consequence of Mrs. Gerber's Lemma due to Wyner and Ziv [24, Corollary 4]:

Suppose (X, W) is a pair of random variables, where X takes values in $\{0,1\}^N$ and $H[X \mid W] = Nv$. Let $Z \in \{0,1\}^N$ be sequence of N independent bits, each taking the value 1 with probability ε; let Z be independent of (X, W). Let $Y = X \oplus Z$. Then,

$$H[Y \mid W] \geqslant Nh(\varepsilon * h^{-1}(v)),$$

where h is the binary entropy function and $\varepsilon * v = \varepsilon(1-v) + (1-\varepsilon)v$. Note that $h(\varepsilon * h^{-1}(v)) \geqslant 1 - (1-v)(1-2\varepsilon)^2$ (see, for example, [7]).

We take $W = g_A(X)$ in the above statement; then, $H[X \mid W] = H[X \mid g_A(X)] = N - H[g_A(X)]$. So, we set $v = 1 - H[g_A(X)]/N$ and conclude that $H[Y \mid g_A(X)] \geqslant N - (1-2\varepsilon)^2 H[g_A(X)]$. Thus, $I[g_A(X) : Y] = H[Y] - H[Y \mid g_A(X)] \leqslant (1-2\varepsilon)^2 H[g_A(X)]$.

(b) We first derive a version of Shearer's lemma. Let $\mathsf{sgn}(Y)$ be the erasure pattern of Y, that is, a sequence in $\{0,1\}^N$, where the 0s correspond to erasures.

$$\begin{aligned}
H[Y \mid \mathsf{sgn}(Y), g_A(X) = z] &= \mathop{\mathbb{E}}_{\sigma}[H[Y \mid \mathsf{sgn}(Y) = \sigma, g_A(X) = z]] \\
&= \mathop{\mathbb{E}}_{\sigma}\left[\sum_{i:\sigma_i=1} H[X_i \mid (X_j : j < i, \sigma_j = 1), g(X) = z]\right] \\
&\geqslant \mathop{\mathbb{E}}_{\sigma}\left[\sum_{i:\sigma_i=1} H[X_i \mid (X_j : j < i), g(X) = z]\right] \\
&= \mathop{\mathbb{E}}_{\sigma}\left[\sum_{i} \mathbf{1}\{\sigma_i = 1\} H[X_i \mid (X_j : j < i), g(X) = z]\right] \\
&= (1-\varepsilon)\sum_{i} H[X_i \mid (X_j : j < i), g(X) = z] \\
&= (1-\varepsilon) H[X \mid g(X) = z].
\end{aligned}$$

Taking expectations of both sides over choices of z, we obtain $H[Y \mid \mathsf{sgn}(Y) g_A(Y)] \geqslant (1-\varepsilon) H[X \mid g_A(X)]$. Then, we have

$$\begin{aligned}
H[Y \mid g_A(X)] &= H[Y\mathsf{sgn}(Y) \mid g_A(X)] \\
&= H[\mathsf{sgn}(Y)] + H[Y \mid \mathsf{sgn}(Y) g_A(X)] \\
&\geqslant h(\varepsilon)N + (1-\varepsilon) H[X \mid g_A(X)]. \qquad (8)
\end{aligned}$$

Thus,

$$\begin{aligned}
I[g_A(X) : Y] &= H[Y] - H[Y \mid g_A(X)] \\
&= h(\varepsilon)N + (1-\varepsilon)N - H[Y \mid g_A(X)] \\
&\leqslant (1-\varepsilon)(N - H[X \mid g_A(X)]) \qquad \text{(using (8))} \\
&= (1-\varepsilon)(H[X] - H[X \mid g_A(X)]) \\
&= (1-\varepsilon) I[X : g_A(X)] = (1-\varepsilon) H[g_A(X)]\,.
\end{aligned}$$

◀

4 The communication lower bounds

We now turn to our lower bounds on communication, formally stated below. Note that these imply the lower bounds claimed in Theorems 1.1 and 1.2.

▶ **Theorem 4.1.** *Let $\gamma, \varepsilon \in [0,1]$ and $k \geqslant 1$ be an integer.*

$$c^{\mathsf{BSC}(N,\varepsilon)}(k', k, 2^{-\gamma k}) \geqslant \left[C(1-\gamma) - 2\sqrt{C(1-C)\gamma}\right] k \qquad \text{where } C = 4\varepsilon(1-\varepsilon); \tag{9}$$

$$c^{\mathsf{BEC}(N,\varepsilon)}(k', k, 2^{-\gamma k}) \geqslant \left[\varepsilon(1-\gamma) - 2\sqrt{\varepsilon(1-\varepsilon)\gamma}\right] k\ . \tag{10}$$

The arguments for the two channels, $\mathsf{BSC}(N,\varepsilon)$ and $\mathsf{BEC}(N,\varepsilon)$, differ only in the choice of the appropriate hypercontractive inequality. We, therefore, first present the common part of the argument. Fix a protocol $\pi \in \Pi(k', k, \eta, R)$, where R is either $\mathsf{BSC}(N,\varepsilon)$ or $\mathsf{BEC}(N,\varepsilon)$. Let $\mathcal{T}$ denote the set of possible transcripts of π; let $t = |\mathcal{T}|$. We will obtain a lower bound on t.

Let $\mathcal{X}, \mathcal{Y}$ denote the domains of X and Y respectively; $\mathcal{X}, \mathcal{Y} = \{0,1\}^N$ for $\mathsf{BSC}(N,\varepsilon)$; $\mathcal{X} = \{0,1\}^N$ and $\mathcal{Y} = \{0,1,?\}^N$ for $\mathsf{BEC}(N,\varepsilon)$. Recall that $g_A(X)$ and $g_B(Y, \pi(X,Y))$ take values in $\mathcal{Z} = \{0,1\}^{k'}$. For $y \in \mathcal{Y}$ and $z \in \mathcal{Z}$, let

$$\beta(z|y) := \Pr[g_A(X) = z \mid Y = y] = \Pr[g_A(X) = z \wedge Y = y] / \Pr[Y = y];$$

let $\mathsf{Success}$ denote the event "$g_A(X) = g_B(Y, \pi(X,Y))$". For $y \in \mathcal{Y}$, let

$$\mathcal{Z}_y = \{g_B(y, \tau) : \tau \in \mathcal{T}\};$$

then, $t_y := |\mathcal{Z}_y| \leqslant t$. On input (x,y), if $g_A(x) \notin \mathcal{Z}_y$, then $\mathsf{Success}$ is impossible. Arrange $z \in \mathcal{Z}_y$ as $z_{y,1}, z_{y,2}, \ldots$ so that $\beta(z_{y,1}|y) \geqslant \beta(z_{y,1}|y) \geqslant \cdots \geqslant \beta(z_{y,t_y}|y)$; let $\beta_{y,i} = \beta(z_{y,i}|y)$.

▶ **Claim 4.2.** *Let $\pi \in \Pi(R, k, \eta)$ be a protocol with t transcripts and let $q > 1$. Then,*

$$\Pr[\mathsf{Success}] \leqslant \mathop{\mathbb{E}}_{Y}\left[\sum_{i=1}^{t_Y} \beta_{Y,i}\right] \leqslant \left(\sum_{z} \mathop{\mathbb{E}}_{Y}\left[\beta(z|Y)^q\right]\right)^{1/q} \cdot t^{1-1/q}. \tag{11}$$

Proof. When Alice sends no message, Bob's best strategy on receiving y is to output the "most likely answer"; so, the probability of $\mathsf{Success}$ is at most β_{y_1}. We now generalize this principle to the case where Bob may base his decision on a transcript. We have

$$\begin{aligned}
\Pr[\mathsf{Success} \mid Y = y] &\leqslant \sum_{\tilde{z} \in \mathcal{Z}_y} \Pr[\mathsf{Success} \wedge g_B(Y, \pi(X,Y)) = \tilde{z} \mid Y = y] \\
&\leqslant \sum_{\tilde{z} \in \mathcal{Z}_y} \Pr[g_A(X) = \tilde{z} \mid Y = y] \\
&= \sum_{\tilde{z} \in \mathcal{Z}_y} \beta(\tilde{z}|y) \leqslant \sum_{i=1}^{t_y} \beta_{y,i},
\end{aligned}$$

where the last inequality holds because $\langle \beta_{y,i} : i = 1, 2, \ldots, t_y \rangle$ are the top t_y values of $\beta(z|y)$. Thus,

$$\begin{aligned}
\Pr[\mathsf{Success}] &\leqslant \mathop{\mathbb{E}}_{Y}\left[\sum_{i=1}^{t_Y} \beta_{Y,i}\right] && (12)\\
&\leqslant \mathop{\mathbb{E}}_{Y}\left[\left(\sum_{i=1}^{t_Y} \beta_{Y,i}^q\right)^{1/q} t_Y^{1-1/q}\right] && \text{(by Hölder's inequality)}\\
&\leqslant \left(\mathop{\mathbb{E}}_{Y}\left[\sum_{i=1}^{t_Y} \beta_{Y,i}^q\right]\right)^{1/q} \cdot t_Y^{1-1/q} && \text{(by Jensen's inequality)} \quad (13)\\
&\leqslant \left(\sum_{z} \mathop{\mathbb{E}}_{Y}\left[\beta(z|Y)^q\right]\right)^{1/q} \cdot t^{1-1/q}. && (14)
\end{aligned}$$

◀

4.1 Hypercontractivity

For functions $\alpha : \mathcal{X} \to \mathbb{R}$ and $\beta : \mathcal{Y} \to \mathbb{R}$, let

$$\|\alpha\|_p = \mathop{\mathbb{E}}_{X}\left[|\alpha(X)|^p\right]^{1/p};$$
$$\|\beta\|_q = \mathop{\mathbb{E}}_{Y}\left[|\beta(Y)|^q\right]^{1/q}.$$

For $z \in \mathcal{Z}$, let $\mathbf{1}_z$ be the indicator random variable $\mathbf{1}[g_A(X) = z]$ and $\beta_z : \mathcal{Y} \to \mathbb{R}$ be defined by $\beta_z(y) = \beta(z|y) = \mathbb{E}[\mathbf{1}_x(X) \mid Y = y]$. Then,

$$\left(\mathop{\mathbb{E}}_{Y}\left[\beta(z|Y)^q\right]\right)^{1/q} = \|\beta_z\|_q = \|\,\mathbb{E}[\mathbf{1}_z(X) \mid Y]\|_q.$$

Using this, we may rearrange inequality (11) and obtain

$$t \geqslant \Pr[\mathsf{Success}]^{q/(q-1)} \left[\sum_z \|\,\mathbb{E}[\mathbf{1}_z(X) \mid Y]\|_q^q\right]^{-1/(q-1)}. \tag{15}$$

Now assume that we have a pair (p, q), $1 \leqslant p < q$, such that for all functions $f : \mathcal{X} \to \mathbb{R}$,

$$\mathbb{E}[f(X) \mid Y]\|_q \leqslant \|f\|_p. \tag{16}$$

Later we will choose an appropriate pair (p, q) depending on the channel. Using (16) with the function $\mathbf{1}_z$, we obtain

$$\begin{aligned}
t &\geqslant \Pr[\mathsf{Success}]^{q/(q-1)} \left[\sum_z \|\mathbf{1}_z\|_p^q\right]^{-1/(q-1)} \\
&= \Pr[\mathsf{Success}]^{q/(q-1)} \left[\sum_z \Pr[g_A(X) = z]^{q/p}\right]^{-1/(q-1)} \\
&\geqslant \Pr[\mathsf{Success}]^{q/(q-1)} \left[\sum_z \Pr[g_A(X) = z] \Pr[g_A(X) = z]^{(q-p)/p}\right]^{-1/(q-1)} \\
&\geqslant \Pr[\mathsf{Success}]^{q/(q-1)} \left[2^{-k(q-p)/p} \sum_z \Pr[g_A(X) = z]\right]^{-1/(q-1)} \qquad (\text{since } H_\infty[g_A(X)] \geqslant k) \\
&\geqslant \Pr[\mathsf{Success}]^{q/(q-1)} \left[2^{k(q-p)/p}\right]^{1/(q-1)}.
\end{aligned}$$

The above argument was general, and applicable for any channel where we can find an appropriate pair (p, q) so that (16) holds. We now specialize the argument to $\mathsf{BSC}(N, \varepsilon)$ and $\mathsf{BEC}(N, \varepsilon)$.

Binary symmetric channel. In this case, we set $q = 1 + \delta$ and $p = 1 + (1 - 2\varepsilon)^2\delta$ [19, Chap. 16]. Then,

$$t \geqslant \Pr[\mathsf{Success}]^{(1+\delta)/\delta} \cdot 2^{4\varepsilon(1-\varepsilon)k/(1+(1-2\varepsilon)^2\delta)}. \tag{17}$$

Binary erasure channel. In this case, for $q = 1 + \delta$, we can take $p = 1 + (1 - \varepsilon)\delta$ [18], and deduce

$$t \geqslant \Pr[\mathsf{Success}]^{(1+\delta)/\delta} \cdot 2^{\varepsilon k/(1+(1-\varepsilon)\delta)}. \tag{18}$$

4.2 The trade-off

Let us fix the success probability at $\eta = 2^{-\gamma k}$ and try to choose δ above so that we obtain the best lower bound on t from (17) and (18).

Binary symmetric channel. Plugging in $\Pr[\mathsf{Success}] = 2^{-\gamma k}$ into (17), we conclude that

$$t \geqslant 2^{r_\gamma^{\mathsf{BSC}(N,\varepsilon)}(\delta)k},$$

where

$$r_\gamma^{\mathsf{BSC}(N,\varepsilon)}(\delta) := \frac{C}{1+(1-C)\delta} - \frac{\gamma}{\delta} - \gamma,$$

and $C = 4\varepsilon(1-\varepsilon)$. We need to choose δ so that $r_\gamma(\delta)$ is maximum. Setting the derivative to zero gives us the optimum choice δ_γ^* for which

$$r_\gamma^{\mathsf{BSC}(N,\varepsilon)}(\delta_\gamma^*) = C(1-\gamma) - 2\sqrt{C(1-C)\gamma} \quad .$$

This justifies our lower bound (9) for $\mathsf{BSC}(N,\varepsilon)$.

Note that at $\gamma = 0$ (success probability constant), this quantity is $4\varepsilon(1-\varepsilon)$. As γ increases, $r_\gamma(\delta_\gamma^*)$ decreases monotonically, and becomes 0 when $\gamma = \varepsilon/(1-\varepsilon)$, at which point we may only conclude that $t \geqslant 1$ (which is consistent with the results of Bogdanov and Mossel [5] for zero communication).

Erasure channel. The calculations are identical. We obtain

$$t \geqslant 2^{r_\gamma^{\mathsf{BEC}(N,\varepsilon)}(\delta)k},$$

where

$$r_\gamma^{\mathsf{BEC}(N,\varepsilon)}(\delta) := \frac{\varepsilon}{1+(1-\varepsilon)\delta} - \frac{\gamma}{\delta} - \gamma.$$

Fixing γ, we find the optimum value δ_γ^* for δ, such that

$$r_\gamma^{\mathsf{BEC}(N,\varepsilon)}(\delta_\gamma^*) = \varepsilon(1-\gamma) - 2\sqrt{\varepsilon(1-\varepsilon)\gamma}.$$

This justifies our lower bound (10) for $\mathsf{BEC}(N,\varepsilon)$. When $\gamma = (1-\sqrt{1-\varepsilon})/(1+\sqrt{1-\varepsilon})$, we obtain $r_\gamma^{\mathsf{BEC}(N,\varepsilon)}(\delta_\gamma^*) = 0$; in the next section we will show that there is indeed a zero communication protocol of $\mathsf{BEC}(N,\varepsilon)$ that succeeds with probability close to $2^{-(1-\sqrt{1-\varepsilon})k/(1+\sqrt{1-\varepsilon})}$.

5 Communication protocols

Our protocols are similar to the protocol of Bogdanov and Mossel [5]. We first recall their protocol. Let $Z = \{0,1\}^k$. Alice and Bob use an affine subspace of $\mathbb{F}_2^n$ (where $\mathbb{F}_2 = \{0,1\}$ is the field with two elements) with 2^k vectors $\mathbf{v} = (v_z : z \in \{0,1\}^k)$. We will assume that this subspace is constructed at random, by the following process: pick k linearly independent vectors $w_1, w_2, \ldots, w_k$ uniformly at random and another random vector $w_0 \in \{0,1\}^N$; then set

$$v_z = w_0 + \sum_{i=1}^{k} z_i w_i.$$

Note, in particular, that if $z, z' \in \{0,1\}^k$ and $z \neq z'$, then $(v_z, v_{z'})$ ranges uniformly over $\{0,1\}^N \times \{0,1\}^N$.

On receiving $X \in \{0,1\}^n$, Alice's output $g_A(X)$ will be the $z \in Z$ for which v_z is closest to X. To break ties, the following rule is used. Fix a total ordering $\preceq$ on $\{0,1\}^N$ such that if the Hamming weight of x is less than the Hamming weight of x', then $x \preceq x'$. Then, $g_A(x) = z$ for which $x + v_z$ is the smallest with respect to $\preceq$. For this function, Bogdanov and Mossel [5] show the following.

▶ **Lemma 5.1.** *For all $z \in \{0,1\}^k$, we have $\Pr_X[f(X) = z] = 2^{-k}$.*

In the original protocol of Bogdanov and Mossel, Bob uses the same function as Alice to produce his output. We extend the above protocol, allowing Alice to send a short message to Bob. Fix a function $\chi : Z \to \{0,1\}^{ck}$ such that $|\chi^{-1}(\alpha)| = 2^{(1-c)k}$ for all $\alpha \in \{0,1\}^{ck}$. Alice's message to Bob is then $m = \chi(g_A(X))$. On receiving the message m, Bob's output is $z \in \chi^{-1}(m)$ for which v_z agrees most with Y (breaking ties arbitrarily).

It will be convenient to state our proofs using $\{+1,-1\}$ instead of $\{0,1\}$; so we assume that the vectors v_z and the random string X take values in $\{+1,-1\}^N \subseteq \mathbb{R}^N$. If the channel is $\mathsf{BSC}(\varepsilon)$, then we will assume that $Y \in \{+1,-1\}^N$; if the channel is $\mathsf{BEC}(\varepsilon)$, then we will assume that $Y \in \{+1,-1,0\}^N$, where 0 corresponds to erasures. Also, we will assume that $\varepsilon \neq 0$, for Alice and Bob have identical strings and they can just out the first k bits.

5.1 Agreement distillation protocol for BSC(ε)

We fix $\gamma > 0$, and describe a protocol with low communication that achieves success probability $2^{-\gamma k - o(k)}$. We will do the computation assuming that the affine space of vectors $\mathbf{v}$ is chosen at random. The overall success probability then is averaged over the random choices of the affine subspace. Clearly, there is a choice of an affine subspace where the success probability is at least this average.

Fix $z \in Z$. Note that the quantity $X \cdot v_z = \sum_i X[i] v_z[i]$ is then a sum of N independent random variables taking values in $\{+1,-1\}$, such that $\mathbb{E}[X \cdot v_z] = 0$ and $\mathbf{var}[X \cdot v_z] = N$. To estimate the probabilities, we will assume that N is large and use the normal approximation. Let

$$\varphi(r) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{r^2}{2}\right);$$
$$\Phi^c(r) = \int_r^\infty \varphi(x)\mathrm{d}x.$$

▶ **Theorem 5.2** (Berry-Esseen theorem [12, Sec. XVI.5, Theorem 2])**.** *Let $S = \xi_1 + \xi_2 + \cdots + \xi_N$, where the ξ_i are independent random variables. Suppose $\mu_i = \mathbb{E}[\xi_i]$, $\sigma_i^2 = \mathbf{var}[\xi_i]$ and $\tau_i = \mathbb{E}[|\xi_i - \mu_i|^3]$. Let $\mu = \mathbb{E}[S] = \sum_i \mu_i$ and $\sigma^2 = \mathbf{var}[S] = \sum_i \sigma_i^2$ and $\tau = \sum_i \tau_i$. Then,*

$$|\Pr[S \geqslant \mu + r\sigma] - \Phi^c(r)| \leqslant \frac{6\tau}{\sigma^3}. \tag{19}$$

In all our applications $\sigma^2 = \Theta(N)$ (the constant depends on ϵ) and $\tau \leqslant N$; thus, the right hand side is $O(1/\sqrt{N})$, where the implicit constant depends only on ϵ and is positive if $\epsilon \in (0,1)$ is positive. In particular, using standard estimates for $\Phi^c(r)$ (see, for example, [8]), we conclude that for all $r > 0$ and all large enough N

$$\frac{r^2}{r^2+1}\varphi(r) < \Pr[S \geqslant \mu + r\sigma] < \varphi(r) \tag{20}$$

Thus, for all large N, one has

$$\varphi(r)\left(1-O(r^{-2})\right) < \Pr[X \cdot v_z \geqslant r\sqrt{N}] < \varphi(r)\ .$$

Note the following behavior of φ when its argument is scaled:

$$\varphi(\alpha r) = \frac{1}{\alpha(\sqrt{2\pi}r)^{1-\alpha^2}}\varphi(r)^{\alpha^2}. \tag{21}$$

Let

$$\eta = 2\varepsilon - \sqrt{4\varepsilon(1-\varepsilon)\gamma}.$$

For $z \in \{0,1\}^k$, let

$$A_z := \left\{x \in \{+1,-1\}^n : v_z \cdot x \geqslant r\sqrt{N}\right\};$$
$$B_z := \left\{x \in \{+1,-1\}^n : v_z \cdot x \geqslant (1-\eta)r\sqrt{N}\right\}.$$

Fix $r = \Theta(\sqrt{k})$, such that for all large enough N

$$2^{-(k+1)} \leqslant \mu(A_z) \leqslant 2^{-(k+1)}\left(1+O(1/k)\right)\ . \tag{22}$$

Consider the following events for $z \in \{0,1\}^k$.

$$\mathcal{E}_1(z) := (X,Y) \in A_z \times B_z;$$
$$\mathcal{E}_2(z) := \forall z' \neq z : X \notin A_{z'};$$
$$\mathcal{E}_3(z) := \forall z' \neq z\,(\chi(z) = \chi(z')) : Y \notin B_{z'}.$$

$\mathcal{E}_1(z)$ and $\mathcal{E}_2(z)$ ensure that Alice outputs z; $\mathcal{E}_1(z)$ and $\mathcal{E}_3(z)$ ensure that Bob outputs z; thus, if all three events hold, then Alice and Bob both output the string z. Thus,

$$\Pr[\mathsf{Success}] \geqslant \sum_z \Pr[\mathcal{E}_1(z)]\left(1 - \Pr[\overline{\mathcal{E}_2(z)} \mid \mathcal{E}_1(z)] - \Pr[\overline{\mathcal{E}_3(z)} \mid \mathcal{E}_1(z)]\right). \tag{23}$$

We will estimate the probabilities appearing on the right separately. First, we have

$$\Pr[\mathcal{E}_1(z)] = \Pr[X \in A_z] \cdot \Pr[Y \in B_z \mid X \in A_z]. \tag{24}$$

For our choice of r (see (22)), $\Pr[X \in A_z] \geqslant 2^{-(k+1)}$. To compute the second factor, fix $\mathbf{v}$ (the affine space of 2^k vectors) and $x \in A_z$; say $x \cdot v_z = r'\sqrt{N}$ for some $r' \geqslant r$. Now, $Y \cdot v_z$ is the sum of N independent random variables taking values in $\{+1,-1\}$, such that $\mathbb{E}[Y \cdot v_z] = (1-2\varepsilon)r'\sqrt{N}$ and $\mathbf{var}[Y \cdot v_z] = 4\varepsilon(1-\varepsilon)N$. Thus,

$$\begin{aligned}
\Pr[Y \cdot v_z \geqslant (1-\eta)r\sqrt{N} \mid X = x] &\geqslant \varphi\left(\frac{(1-\eta)r - (1-2\varepsilon)r'}{\sqrt{4\varepsilon(1-\varepsilon)}}\right)(1-O(1/k)) \\
&\geqslant \varphi\left(\frac{(1-\eta)r - (1-2\varepsilon)r}{\sqrt{4\varepsilon(1-\varepsilon)}}\right)(1-O(1/k)) \\
&\qquad \text{(since } \varphi(r) \text{ is decreasing)} \\
&= \varphi\left(\frac{(2\varepsilon-\eta)r}{\sqrt{4\varepsilon(1-\varepsilon)}}\right)(1-O(1/k)) \\
&= \varphi\left(\sqrt{\gamma}r\right)(1-O(1/k)) \\
&\geqslant \frac{1}{\sqrt{\gamma}(\sqrt{2\pi}r)^{1-\gamma}}2^{-\gamma(k+1)}(1-O(1/k)).
\end{aligned} \tag{25}$$

Using these in (24), we obtain

$$\begin{aligned}\Pr[\mathcal{E}_1(z)] &= \Pr[X \in A_z] \cdot \Pr[Y \in B_z \mid X \in A_z] \\ &\geqslant \frac{1}{\sqrt{\gamma}(\sqrt{2\pi}r)^{1-\gamma}} 2^{-(\gamma+1)(k+1)}(1 - O(1/k)). \end{aligned} \tag{26}$$

Recall that if $z \neq z'$, then as $\mathbf{v}$ varies, v_z and $v_{z'}$ vary uniformly over all pairs of distinct vectors in $\{+1, -1\}^N$. It follows that

$$\begin{aligned}\Pr[\overline{\mathcal{E}_2(z)} \mid \mathcal{E}_1(z)] &\leqslant \sum_{z' : z' \neq z} \Pr[0^N \in A_{z'}] \\ &\leqslant (2^k - 1) \cdot 2^{-(k+1)}(1 + O(k^{-1})) \\ &\leqslant \frac{1}{2}(1 + O(k^{-1})). \end{aligned} \tag{27}$$

Similarly, we have

$$\begin{aligned}\Pr[\overline{\mathcal{E}_3(z)} \mid \mathcal{E}_1(z)] &\leqslant \sum_{z' : \chi(z) = \chi(z'), z' \neq z} \Pr[Y \in B_{z'}] \\ &\leqslant 2^{(1-c)k} \varphi((1-\eta)r) \\ &\leqslant 2^{(1-c)k} \frac{1}{(1-\eta)(\sqrt{2\pi}r)^{\eta(2-\eta)}} 2^{-(1-\eta)^2(k+1)}. \quad \text{(see (21) above)} \end{aligned} \tag{28}$$

Thus, if $c \geqslant 1 - (1-\eta)^2 = C(1-\gamma) - 2\sqrt{C(1-C)\gamma}$ where $C = 4\varepsilon(1-\varepsilon)$, then this quantity is at most $\frac{1}{4}$ (say) for all large k. It follows from (23), (26), (27) and (28) that

$$\begin{aligned}\Pr_{\mathbf{v},X,Y}[\mathsf{Success}] &\geqslant \sum_z \frac{1}{\sqrt{\gamma}(\sqrt{2\pi}r)^{1-\gamma}} 2^{-(\gamma+1)(k+1)} \left(1 - \frac{1}{2} - \frac{1}{4}\right)(1 - O(k^{-1})) \\ &= 2^{-\gamma k - O(\log \gamma k)} \\ &= 2^{-\gamma k(1+o(1))}. \end{aligned}$$

Thus, there exists a choice of the subspace $\mathbf{v}$ such that Alice and Bob succeed with probability at least $2^{-\gamma k(1+o(1))}$.

Constant probability of success. The above argument, was carried out with $\gamma > 0$ a constant, so that it yielded agreement with probability $2^{-\gamma k(1+o(1))}$. We may, in fact, set $\gamma = 1/r^2 = \Theta(1/k)$ in the above argument, and conclude that with communication $ck \approx (C(1-\gamma) - 2\sqrt{C(1-C)\gamma})k = 4\varepsilon(1-\varepsilon)k - \Theta(\sqrt{k})$, we obtain $\Pr_{\mathbf{v},X,Y}[\mathsf{Success}] = \Omega(1)$.

5.2 Agreement distillation protocol for BEC(ε)

The calculations are similar to the one we used above. We fix r and A_z as before. However, this time we set $\eta = \varepsilon - \sqrt{\varepsilon(1-\varepsilon)\gamma}$ and let

$$B_z := \left\{ x \in \{+1, -1\}^n : v_z \cdot x \geqslant (1-\eta) r \sqrt{N} \right\}.$$

We define events $\mathcal{E}_1(z)$, $\mathcal{E}_2(z)$ and $\mathcal{E}_3(z)$ as before, and observe that

$$\Pr[\mathsf{Success}] \geqslant \sum_z \Pr[\mathcal{E}_1(z)](1 - \Pr[\overline{\mathcal{E}_2(z)} \mid \mathcal{E}_1(z)] - \Pr[\overline{\mathcal{E}_3(z)} \mid \mathcal{E}_1(z)]). \tag{29}$$

continues to hold. To estimate the first factor, we expand it as before and obtain

$$\Pr[\mathcal{E}_1(z)] = \Pr[X \in A_z] \cdot \Pr[Y \in B_z \mid X \in A_z].$$

$\Pr[X \in A_z] \geqslant 2^{-(k+1)}$. As before, for each fixed $x \in A_z$ (such that $x \cdot v_z = r'\sqrt{N}$, $r' \geqslant r$), we view $Y \cdot v_z$ as a sum of N independent random variables, each taking values in either $\{0, +1\}$ or $\{0, -1\}$; in particular, $\mathbb{E}[Y \cdot v_z] = (1-\varepsilon)r'\sqrt{N}$ and $\mathbf{var}[Y \cdot v_z] = \varepsilon(1-\varepsilon)N$. Thus,

$$\begin{aligned}\Pr[Y \cdot v_z \geqslant (1-\eta)r \mid X = x] &\geqslant \varphi\left(\frac{(1-\eta)r - (1-\varepsilon)r'}{\sqrt{\varepsilon(1-\varepsilon)}}\right)(1 - O(k^{-1})) \\ &\geqslant \frac{1}{\sqrt{\gamma}(\sqrt{2\pi}r)^{1-\gamma}} 2^{-\gamma(k+1)}(1 - O(k^{-1})).\end{aligned}$$

using calculations identical to those leading to (25). We finally have the following lower bound for the first factor of (29).

$$\begin{aligned}\Pr[\mathcal{E}_1(z)] &= \Pr[X \in A_z] \cdot \Pr[Y \in B_z \mid X \in A_z] \\ &\geqslant \frac{1}{\sqrt{\gamma}(\sqrt{2\pi}r)^{1-\gamma}} 2^{-(\gamma+1)(k+1)}(1 - O(k^{-1})). \qquad (30)\end{aligned}$$

Calculations that lead to

$$\Pr[\overline{\mathcal{E}_2(z)} \mid \mathcal{E}_1(z)] \leqslant \frac{1}{2}(1 + O(k^{-1})) \qquad (31)$$

remain the same.

Finally, we consider $\mathcal{E}_3(z)$. First, we observe that since N is large, we may assume that with probability tending to 1, the number of ones in Y is $(1-\varepsilon)N \pm N^{3/4}$ (say), even when conditioning on $\mathcal{E}_1$. Now, the pair $(v_z, v_{z'})$ is uniformly distributed over all possible pairs of distinct vectors. So, we will fix v_z, and assume that $v_{z'}$ is uniformly distributed in $\{+1, -1\}^N$ (that it cannot be v_z can be overlooked). Fix y with say $\ell = (\varepsilon + N^{-1/4})N = \varepsilon' N$ zeroes. Then, $Y \cdot v_{z'}$ is the sum of $(1-\varepsilon')N$ independent random variables, each taking values uniformly in $\{+1, -1\}$. In particular, $\mathbb{E}[Y \cdot v_{z'}] = 0$ and $\mathbf{var}[Y \cdot v_{z'}] = (1-\varepsilon')N$. Then,

$$\begin{aligned}\Pr[\overline{\mathcal{E}_3(z)} \mid \mathcal{E}_1(z)] &\leqslant \sum_{z' : \chi(z) = \chi(z'), z' \neq z} \Pr[Y \in B_{z'}] \\ &\leqslant 2^{(1-c)k}\varphi\left(\frac{(1-\eta)}{\sqrt{1-\varepsilon'}}r\right)\left(1 + O(k^{-1})\right) \\ &\leqslant 2^{(1-c)k}\frac{1}{(1-\eta)(\sqrt{2\pi}r)^{1-\frac{(1-\eta)^2}{(1-\varepsilon')}}} 2^{-\frac{(1-\eta)^2(k+1)}{(1-\varepsilon')}}\left(1 + O(k^{-1})\right), \qquad (32)\end{aligned}$$

where in the last step we used (21). Thus, if $c \geqslant 1 - (1-\eta)^2/(1-\varepsilon') = \varepsilon(1-\gamma) - 2\sqrt{\varepsilon(1-\varepsilon)\gamma}$, then this quantity is at most $\frac{1}{4}$ (say) for all large k. It follows from (29), (30), (31) and (32) that

$$\begin{aligned}\Pr_{\mathbf{v},X,Y}[\mathsf{Success}] &\geqslant \sum_z \frac{1}{\sqrt{\gamma}(\sqrt{2\pi}r)^{1-\gamma}} 2^{-(\gamma+1)(k+1)}\left(1 - \frac{1}{2} - \frac{1}{4}\right)(1 - O(k^{-1})) \\ &= 2^{-\gamma k - O(\log \gamma k)}.\end{aligned}$$

We may, as before, fix a choice of $\mathbf{v}$ such that Alice and Bob succeed with probability at least $2^{-\gamma k(1+o(1))}$.

Constant probability of success. Again, we may set $\gamma = 1/r^2 = \Theta(1/k)$ in the above argument, and conclude that with communication $\varepsilon k - \Theta(\sqrt{k})$, we obtain $\Pr_{\mathbf{v},X,Y}[\mathsf{Success}] = \Omega(1)$.

6 Open problems

Our work raises a number of intriguing open questions, such as:

- Is there a protocol for general channels whose communication, for agreeing on a k-bit random string with constant probability, approaches $s^*(Y;X)k$? Here $s^*(Y;X)$ is the channel parameter defined in (1).
- We considered protocols where the shared random string was a function $g_A(X)$ of Alice's input X. What can we achieved by a general multi-round communication protocol, where the shared random string can depend on both X and Y? Can we do better than the lower bounds we established, or do the lower bounds continue to hold in this (seemingly) more powerful model?
- The setup for $\mathsf{BEC}(\varepsilon)$ is not symmetric between Alice and Bob. What can be done if Alice and Bob switch roles, and the shared randomness should be a function of Y? What are the possible trade-offs in the symmetric setup where X and Y are the independent outputs of $\mathsf{BEC}(\varepsilon)$ on a common random string $Z \in \{0,1\}^N$?

Acknowledgments. We thank the Simons Institute for the Theory of Computing, Berkeley, where most of this work was done. We are grateful to T. S. Jayram and Chandra Nair for several illuminating discussions related to this work.

References

1 Rudolf Ahlswede and Imre Csiszár. Common randomness in information theory and cryptography – part II: CR capacity. *IEEE Transactions on Information Theory*, 44(1):225–240, 1998.

2 Venkat Anantharam, Amin Aminzadeh Gohari, Sudeep Kamath, and Chandra Nair. On hypercontractivity and the mutual information between boolean functions. In *Proceedings of the 51st Annual Allerton Conference on Communication, Control, and Computing*, pages 13–19, 2013. doi:10.1109/Allerton.2013.6736499.

3 Venkat Anantharam, Amin Aminzadeh Gohari, Sudeep Kamath, and Chandra Nair. On maximal correlation, hypercontractivity, and the data processing inequality studied by Erkip and Cover. *CoRR*, abs/1304.6133, 2013. URL: http://arxiv.org/abs/1304.6133.

4 Venkat Anantharam, Amin Aminzadeh Gohari, Sudeep Kamath, and Chandra Nair. On hypercontractivity and a data processing inequality. In *2014 IEEE International Symposium on Information Theory*, pages 3022–3026, 2014. doi:10.1109/ISIT.2014.6875389.

5 Andrej Bogdanov and Elchanan Mossel. On extracting common random bits from correlated sources. *IEEE Transactions on Information Theory*, 57(10):6351–6355, 2011.

6 Clément Louis Canonne, Venkatesan Guruswami, Raghu Meka, and Madhu Sudan. Communication with imperfectly shared randomness. In *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science*, pages 257–262, 2015. doi:10.1145/2688073.2688099.

7 Venkat Chandar and Aslan Tchamkerten. Most informative quantization functions. In *Proceedings of ITA Workshop*, 2014. Available at http://perso.telecom-paristech.fr/ tchamker/CTAT.pdf.

8 John D. Cook. *Upper and lower bounds for the normal distribution function*, 2009. http://www.johndcook.com/normalbounds.pdf.

9 Thomas A. Courtade and Gowtham R. Kumar. Which Boolean functions maximize mutual information on noisy inputs? *IEEE Transactions on Information Theory*, 60(8):4515–4525, 2014. doi:10.1109/TIT.2014.2326877.

10 Elza Erkip. *The efficiency of information in investment*. PhD thesis, Stanford University, 1996.

11 Elza Erkip and Thomas M. Cover. The efficiency of investment information. *IEEE Transactions on Information Theory*, 44(3):1026–1040, 1998. doi:10.1109/18.669153.

12 William Feller. *An Introduction to Probability Theory and Its Applications*, volume 2. John Wiley and Sons, 1971.

13 Péter Gács and János Körner. Common information is far less than mutual information. *Problems of Control and Information Theory*, 2(2):119–162, 1972.

14 Guy Kindler, Ryan O'Donnell, and David Witmer. Remarks on the most informative function conjecture at fixed mean. *CoRR*, abs/1506.03167, 2015. URL: http://arxiv.org/abs/1506.03167.

15 Konstantin Makarychev and Yury Makarychev. Chain independence and common information. *IEEE Transactions on Information Theory*, 58(8):5279–5286, 2012.

16 Elchanan Mossel and Ryan O'Donnell. Coin flipping from a cosmic source: On error correction of truly random bits. *Random Struct. Algorithms*, 26(4):418–436, 2005. doi:10.1002/rsa.20062.

17 Elchanan Mossel, Ryan O'Donnell, Oded Regev, Jeffrey Steif, and Benny Sudakov. Non-interactive correlation distillation, inhomogeneous Markov chains, and the reverse bonami–beckner inequality. *Israel J. Math.*, 154:299–336, 2006.

18 Chandra Nair and Yannan Wang. Evaluating hypercontractivity parameters using information measures. Manuscript, 2015.

19 Ryan O'Donnell. *Analysis of Boolean Functions*. Cambridge University Press, 2014. URL: http://www.cambridge.org/de/academic/subjects/computer-science/algorithmics-complexity-computer-algebra-and-computational-g/analysis-boolean-functions.

20 Or Ordentlich, Ofer Shayevitz, and Omri Weinstein. Dictatorship is the most informative balanced function at the extremes. *Electronic Colloquium on Computational Complexity (ECCC)*, 22:84, 2015. URL: http://eccc.hpi-web.de/report/2015/084.

21 Andrei Romashchenko. Pairs of words with nonmaterializable mutual information. *Problems of Information Transmission*, 36(1):1–18, 2000.

22 Alex Samorodnitsky. The "most informative boolean function" conjecture holds for high noise. *CoRR*, abs/1510.08656, 2015.

23 Alex Samorodnitsky. On the entropy of a noisy function. *Electronic Colloquium on Computational Complexity (ECCC)*, 22:129, 2015.

24 Aaron D. Wyner and Jacob Ziv. A theorem on the entropy of certain binary sequences and applications: Part 1. *IEEE Transactions on Information Theory*, 19(6):769–772, 1973. doi:10.1109/TIT.1973.1055107.

25 Ke Yang. On the (im)possibility of non-interactive correlation distillation. *Theor. Comput. Sci.*, 382(2):157–166, 2007. doi:10.1016/j.tcs.2007.03.007.

26 Lei Zhao and Yeow-Khiang Chia. The efficiency of common randomness generation. In *Proceedings of 49th Annual Allerton Conference on Communication, Control, and Computing*, pages 944–950, 2011.

New Extractors for Interleaved Sources*

Eshan Chattopadhyay[1] and David Zuckerman[2]

1 **University of Texas at Austin, Austin, USA**
eshanc@cs.utexas.edu
2 **University of Texas at Austin, Austin, USA**
diz@cs.utexas.edu

Abstract

We study how to extract randomness from a C-interleaved source, that is, a source comprised of C independent sources whose bits or symbols are interleaved. We describe a simple approach for constructing such extractors that yields:

- For some $\delta > 0, c > 0$, explicit extractors for 2-interleaved sources on $\{0,1\}^{2n}$ when one source has min-entropy at least $(1-\delta)n$ and the other has min-entropy at least $c \log n$. The best previous construction, by Raz and Yehudayoff [36], worked only when both sources had entropy rate $1-\delta$.
- For some $c > 0$ and any large enough prime p, explicit extractors for 2-interleaved sources on $[p]^{2n}$ when one source has min-entropy rate at least .51 and the other source has min-entropy rate at least $(c \log n)/n$.

We use these to obtain the following applications:

- We introduce the class of any-order-small-space sources, generalizing the class of small-space sources studied by Kamp et al. [22]. We construct extractors for such sources with min-entropy rate close to 1/2. Using the Raz-Yehudayoff construction would require entropy rate close to 1.
- For any large enough prime p, we exhibit an explicit function $f : [p]^{2n} \to \{0,1\}$ such that the randomized best-partition communication complexity of f with error $1/2 - 2^{-\Omega(n)}$ is at least $.24n \log p$. Previously this was known only for a tiny constant instead of .24, for $p = 2$ [36].
- We introduce non-malleable extractors in the interleaved model. For any large enough prime p, we give an explicit construction of a weak-seeded non-malleable extractor for sources over $[p]^n$ with min-entropy rate .51. Nothing was known previously, even for almost full min-entropy.

1998 ACM Subject Classification F. Theory of Computation

Keywords and phrases extractors,derandomization,explicit construction

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.7

1 Introduction

Extracting truly random bits from various naturally-arising weak random sources is a major area of study in computer science, and has applications in various areas such as cryptography, coding theory, communication complexity, and distributed computing. An extractor is defined to be a procedure that takes input from a weak random source and outputs a distribution that is close to uniform.

Von Neumann [41] initiated the study of weak random sources, showing how to extract from a source with independent and biased bits. Various other models of weak random

* This work was partially supported by NSF Grant CCF-1218723.

sources were considered [3, 37, 11], but it was realized that devising such extractors was impossible for any general class of weak random sources lacking significant independence between different parts.

To get around this difficulty, Nisan and Zuckerman [33] introduced the notion of a seeded extractor, which uses a small number of uniformly random bits to extract randomness from a weak source X. The min-entropy of a weak source X is a standard way of measuring of the amount of randomness in X, and is defined as $H_\infty(X) = \min_{s \in \text{support}(X)} \{1/\log(\Pr[X = s])\}$. The min-entropy rate of X supported on $\{0,1\}^n$ is given by $H_\infty(X)/n$. By a long line of work ending with [31, 15, 21], we now have explicit seeded extractors with almost optimal parameters.

In recent years, there has been renewed interest in the original problem of constructing seedless extractors for weak random sources. In particular, a line of work has focused on devising seedless extractors that takes input C independent weak sources $X_1, \ldots, X_C$, and outputs a distribution close to uniform. This problem was originally considered by Chor and Goldreich [10], who showed how to extract from two independent sources (on $\{0,1\}^n$) each with min-entropy at least $(\frac{1}{2} + \delta)n$. Such extractors are called two-source extractors.

However, there was no progress on this result for around 20 years until the work of Bourgain [4], who achieved a small improvement over [10], and showed how to extract from two independent sources each with min-entropy $0.49n$, based on techniques from the area of additive combinatorics. Subsequently, Raz [35] gave an explicit two source extrator, with one source having min-entropy at least $(\frac{1}{2} + \delta)n$ and the other source having poly-logarithmic min-entropy at least $O(\log n)$. Finally, the authors recently constructed two-source extractors for polylogarithmic min-entropy with one bit output [9]. Subsequently, Li [30] improved the output length to $\Omega(k)$ bits.

1.1 Interleaved Sources

Raz and Yehudayoff [36] introduced a natural generalization of the class of independent sources, which we call interleaved sources. We formally define this class of sources.

Notation. Let $[n]$ denote the set $\{1, \ldots, n\}$. For any string $s \in [R]^n$ and $i \in [n]$, let s_i denote the symbol in the ith coordinate of s. For any permutation $t : [n] \to [n]$, define the string $w = (s)_t \in [R]^n$ such that $w_i = s_{t(i)}$ for $i = 1, \ldots, n$. For distributions D_1 and D_2, we use $|D_1 - D_2|$ to denote the statistical (or variation) distance. We use $D_1 \approx_\epsilon D_2$ to denote that $|D_1 - D_2| \leq \epsilon$. See Section 3 for more preliminaries. Let $\circ$ denote standard string concatenation.

▶ **Definition 1.1** (Interleaved Sources)**.** Let $X_1, \ldots, X_C$ be arbitrary independent sources on $[R]^n$ and let $t : [Cn] \to [Cn]$ be any permutation. Then $Z = (X_1 \circ \ldots \circ X_C)_t$ is a C-interleaved source.

Such sources can arise naturally is when the independent sources are communicated remotely to an extractor and packets of bits from different sources arrive in a fixed but unknown order. We show that extractors for interleaved sources can be used to construct extractors for certain samplable sources, thus extending the line of work initiated by Trevisan and Vadhan [40]. We discuss this in Section 1.2. Further, Raz and Yehudayoff [36] showed that such extractors have applications in communication complexity (see Section 1.3) and proving lower bounds for arithmetic circuits.

Previous Results

The only known construction of an extractor for the class of interleaved sources is due to Raz and Yehudayoff [36]. They constructed extractors for 2-interleaved sources on $\{0,1\}^{2n}$ when both sources have min-entropy rate at least $1-\beta$, with output length $\Omega(\beta n)$ and exponentially small error.

The constant β in the result of [36] is tiny and arises from a multilinear exponential sum estimate from [5] (which is based on sum-product estimates on finite fields [6, 26]). Thus, the only known construction required both the sources to have almost full min-entropy. Moreover, their analysis requires estimating a non-trivial exponential sum, and is quite involved.

Our Results

We develop a simple technique that yields explicit extractors that work for lower min-entropy rates. In particular, our method yields explicit extractors for min-entropy rate 0.51 for two interleaved sources, when the sources are over a finite field of large enough (constant) characteristic.

We show how to convert any two-source extractor that is a function of the sum of its inputs into an extractor for a 2-interleaved source. Our method of converting a two-source extractor into an extractor for interleaved sources is based on explicit constructions of certain combinatorial sets, which we call (r,s)-spanning sets. These spanning sets are essentially subspace-evasive sets with different parameters than studied earlier (see Section 2.1 for more details). It turns out that the columns of parity check matrices of linear codes with good erasure list-decodability form spanning sets with good parameters. We discuss this in detail later.

Next, we observe that an existing two-source extractor from [10] is a function of the sum of the inputs. This leads to our construction of an extractor for 2-interleaved sources with one source having min-entropy at least $(1-\alpha)n$ and the other source having min-entropy at least $\lambda \log n$ (for some $\alpha, \lambda > 0$). In particular, we have the following theorem.

▶ **Theorem 1.2.** *For some $\delta > 0$ and any $\lambda > 0$, there exists an explicit function* $\mathrm{ext} : \{0,1\}^{2n} \to \{0,1\}^m$, *$m = \lambda \log n$, such that if X, Y are independent sources on $\mathbb{F}_2^n$ with min-entropy k_1, k_2 respectively satisfying $k_1 > (1-\delta)n$ and $k_2 > 35 \max\{\log n, m\}$, $t : [2n] \to [2n]$ is any permutation, then*

$$|\mathrm{ext}((X \circ Y)_t) \circ X - U_m \circ X| = n^{-\Omega(1)}.$$

Next, we show that for any large enough constant prime p, if the 2-interleaved source is on $[p]^{2n}$, we can extract when one source has min-entropy rate at least 0.51 and the other source has min-entropy rate at least $c \log n/n$.

▶ **Theorem 1.3.** *There exists $c > 0$ such that for any $\delta, \lambda > 0$ and any prime $p > 2^{\frac{c}{\delta}}$, there exists an explicit function* $\mathrm{ext}_p : \mathbb{F}_p^{2n} \to \{0,1\}^m$, *$m = \lambda \log n$, such that if X and Y are independent sources on $\mathbb{F}_p^n$ with min-entropy k_1, k_2 respectively, satisfying $k_1 > (\frac{1}{2} + \delta) n \log p$ and $k_2 > \frac{5}{\delta} \max\{\log n \log p, m\}$, $t : [2n] \to [2n]$ is any injective map, then*

$$|\mathrm{ext}_p((X \circ Y)_t) \circ X - U_m \circ X| = n^{-\Omega(1)}.$$

We give various related constructions achieving different tradeoffs between min-entropy, error, and output length. This is summarized in Table 1.

We show that random sets are (r,s)-spanners with high probability (see Lemma 5.10). By our proof technique, any improved construction of an (r,s)-spanning set matching the

Table 1 Results on Extractors for 2-Interleaved Sources. The setting is as follows: $Z = (X \circ Y)_t$ is an arbitrary 2-interleaved source on $[p]^{2n}$, where X and Y are independent sources on $[p]^n$ (for some prime p) with min-entropy k_1 and k_2 respectively, and $t : [2n] \to [2n]$ is an arbitrary permutation. Let α be a small enough constant and c a large enough constant. Also let $\lambda > 1$ be any constant. We also list the result of [36] in Table 1.

p	k_1	k_2	Output Length	Error	Reference	Remarks
2	$\geq (1-\beta)n$	$\geq (1-\beta)n$	γn, $\gamma < \beta$	$2^{-\Omega(n)}$	[36]	Not strong
2	$\geq (1-\alpha)n$	$\geq 35\lambda \log n$	$\lambda \log n$	$n^{-\alpha}$	Theorem 1.2	Strong in X
2	$\geq (1-\alpha)n$	$\geq 35\lambda \log n$	Output in $\mathbb{Z}_M$, $M = n^\lambda$	$2^{-\Omega(k_2)}$	Theorem 1.3	Strong in X
any $p > 2^{\frac{c}{\delta}}$	$\geq (\frac{1}{2}+\delta)n \log p$	$\geq c_1(\delta,\lambda,p)\log n$	$\lambda \log n$	$n^{-\alpha}$	Theorem 6.5	Strong in X
any $p > 2^{\frac{c}{\delta}}$	$\geq (\frac{1}{2}+\delta)n \log p$	$\geq (\frac{1}{2}+\delta)n \log p$	$\Omega(n)$	$n^{-\alpha}$	Theorem 6.6	Not strong
any $p > 2^{\frac{c}{\delta}}$	$\geq (\frac{1}{2}+\delta)n \log p$	$\geq c_2(\delta,\lambda,p)\log n$	1 bit	$2^{-\Omega(k_2)}$	Theorem 6.7	Strong in X
any $p > 2^{\frac{c}{\delta}}$	$\geq (\frac{1}{2}+\delta)n \log p$	$\geq c_1(\delta,\lambda,p)\lambda \log n$	$\Omega(k_2)$	$2^{-\Omega(k_2)}$	Theorem 6.9	Semi-explicit construction
2	$\geq \gamma n$, any constant γ	$\geq \gamma n$	$\lambda \log n$	$n^{-\alpha}$	Theorem 6.11	Assuming Generalized Paley Graph Conjecture

probabilistic method will yield extractors for 2-interleaved sources on $\{0,1\}^{2n}$ that have essentially the same min-entropy requirement as the standard (non-interleaved) setting.

Subsequent Work

In subsequent work, Chattopadhyay and Li [8] constructed extractors for C-interleaved sources with polylogarithmic min-entropy, for some large enough constant C. However, their results don't apply for $C = 2$, and their construction is more complicated.

1.2 Any-Order-Small-Space-Sources

Trevisan and Vadhan [40] introduced the problem of constructing seedless extractors for the class of samplable sources (the weak random source is generated by an efficient algorithm) and constructed explicit extractors based on some complexity-theoretic assumptions. Subsequently, Kamp et al. [22] introduced a class of samplable sources called small-space sources, where the algorithm generating the source has bounded space. They constructed seedless extractors for such sources with linear min-entropy. Most sources considered previously (for seedless extraction) can be computed in small-space (see [22] for more details). In particular, extractors for small-space sources also extract from bit-fixing sources and symbol-fixing sources, and thus have applications in cryptography [23].

We introduce a natural generalization of small-space sources. For this, we recall the definition of small-space sources from [22].

▶ **Definition 1.4** (Small-Space Sources [22]). A space s source X on $[r]^n$ is generated by a r-way branching program of length n and width 2^s in the following way: The r-way branching program is a layered graph with $n+1$ layers, with each layer containing 2^s vertices and a single start vertex. Each edge is labeled with a variable X_j, a probability value and a

symbol in $[r]$. Further all edges between the ith and $(i+1)$th layer are labelled with the same variable X_i. The output of the source is a random walk beginning at the start vertex, assigning the symbol on the edge to the corresponding variable and finally outputting the generated string.

Note that in the above definition, the variable assigned to an edge is known (for example, all edges between the ith and $(i+1)$th layers have the variable X_i assigned to it). We introduce the natural generalization where the branching program is oblivious but the variable assigned to an edge is unknown. In particular, for an unknown permutation $t : [n] \to [n]$, all edges between the ith and $(i+1)$th layers have the variable $X_{t(i)}$ assigned to it.

We formally define this class of sources.

▶ **Definition 1.5** (Any-Order-Small-Space-Sources)**.** An any-order-space s source X on $[r]^n$ is generated by an r-way branching program of length n and width 2^s and a permutation $t : [n] \to [n]$ in the following way: The r-way branching program is a layered graph with $n+1$ layers and a single start vertex. Each edge is labeled with a variable X_j, a probability value and a symbol in $[r]$. Further all edges between the ith and $(i+1)$th layer are labelled with same variable $X_{t(i)}$. The output of the source is a random walk starting from the start vertex, assigning the symbol on the edge to the corresponding variable and finally outputting the generated string.

Our Results

To construct extractors for the class of any-order-oblivious-small-space sources, we reduce it to the task of extracting from 2-interleaved sources by adapting the technique of [22] to our situation.

Consider an arbitrary any-order-space $s = \delta n/2$ source X on $[p]^n$ (for some constant p) with min-entropy $k = (\frac{1}{2} + \delta)n \log p$. By conditioning on the state of the p-way branching program at the $\frac{n}{2}$th layer, it follows by Lemma 4.7 that X is $2^{-\Omega(n)}$-close to a source $Z = (Y_1 \circ Y_2)_t$, where Y_1 and Y_2 are independent sources on $[p]^{\frac{n}{2}}$ with $\min\{H_\infty(Y_1), H_\infty(Y_2)\} \geq \frac{\delta n \log p}{8}$ and $\max\{H_\infty(Y_1), H_\infty(Y_2)\} \geq (\frac{1}{2} + \frac{\delta}{8})\frac{n \log p}{2}$, and $t : [n] \to [n]$ is a permutation.

It thus follows that all our extractor constructions for 2-interleaved sources also extract from any-order-small-space sources (by splitting the input string into two equal parts and applying the extractor).

Using this reduction, we obtain the first explicit construction of an extractor for any-order-oblivious-small-space sources with min-entropy rate close to $\frac{1}{2}$ (by using the extractor from Theorem 6.5).

▶ **Theorem 1.6.** *There exists $c > 0$ such that for any $\delta \geq 2\delta_1 > 0$ and any prime $p > 2^{\frac{c}{\delta}}$, there exists an explicit function $\text{ext} : [p]^n \to \{0,1\}^m$, $m = O(\log n)$, such that if X is an any-order-oblivious space $s = \delta_1 n$ source on $[p]^n$ with min-entropy $(\frac{1}{2} + \delta)n \log p$, then*

$$|\text{ext}(X) - U_m| = n^{-\Omega(1)}.$$

We note that using our reduction, the extractor from [36] can be used to extract from any-order-small-space sources with min-entropy rate very close to 1.

1.3 Applications to Communication Complexity

Since Yao introduced communication complexity in 1978 Yao [42], there has been an extensive amount of research done on various models of communication (see [27] for formal definitions

and background). We recall the definition of the randomized best-partition communication complexity of an arbitrary function $f : [R]^{2n} \to \{0,1\}$, which generalizes the usual setting where the partition of inputs is known.

Let Alice and Bob be two players who want to collectively compute f following a protocol Π and having access to a common random string r. Fix an arbitrary partition of the set $[2n]$ into 2 subsets of equal size, say S and T. For arbitrary $x, y \in [R]^n$, Alice is given x and Bob receives y and the goal is to compute $f(z)$ with probability at least $1 - \epsilon$, where $z \in [R]^{2n}$ such that $z_S = x$ and $z_T = y$.

For any protocol Π, the randomized communication cost of f with respect to an equi-partition $S, T \subset [2n]$ denoted by $R^{\epsilon}_{\Pi,S,T}(f)$, is defined to be the maximum communication between Alice and Bob over all inputs x, y in the scenario described above. The best-partition communication complexity of f, denoted by $R^{best,\epsilon}(f)$ is defined as:

$$R^{best,\epsilon}(f) = \min_{\Pi} \left\{ \min_{\substack{S,T:|S|=|T|=n,\\ S\cup T=[2n]}} R^{\epsilon}_{\Pi,S,T}(f) \right\}.$$

Lower bounds on the best-partition communication complexity of f implies lower bounds on branching programs computing f [1] and also imply time/space tradeoffs for VLSI circuits [28].

Raz and Yehudayoff [36] proved the following lower bound.

▶ **Theorem 1.7** ([36]). *For some $\beta > 0$, there exists an explicit function $f : \{0,1\}^{2n} \to \{0,1\}$ such that the randomized best-partition communication complexity of f with error $\epsilon = \frac{1}{2} - 2^{-\beta n}$ is at least βn.*

The constant β in the above theorem is, however, extremely small and arises from arguments in additive combinatorics. A similar bound also follows from their work for inputs on $[R]^{2n}$ (for any constant R) and it appears nontrivial to use their techniques to obtain bounds for larger β.

Our Results

We obtain the following result.

▶ **Theorem 1.8.** *There exists $c > 0$ such that for any $\delta, \gamma > 0$ and any prime $p > 2^{\frac{c}{\delta}}$, there exists an explicit function $f : [p]^{2n} \to \{0,1\}$ such that the randomized best-partition communication complexity of f with error $\epsilon = \frac{1}{2} - p^{-\gamma n}$ is at least $(\frac{1}{4} - \delta - \gamma) n \log p$.*

We prove this using a well known technique of lower bounding randomized communication complexity by discrepancy. Our explicit function is the 1-bit extractor constructed in Theorem 6.7. However, we need to analyze the error of the extractor more carefully to obtain the above bound. We prove Theorem 1.8 in Section 8.

1.4 Interleaved-Non-Malleable Extractors

Dodis and Wichs [14] introduced non-malleable extractors, where they showed that explicit constructions of good non-malleable extractors imply almost optimal protocols for privacy amplification, which is a very well studied problem in cryptography. Recently, non-malleable extractors were also used in constructing explicit two-source extractors [9]. We introduce the natural generalization of non-malleable extractors in the interleaved model.

We first recall the definition of a non-malleable extractor.

▶ **Definition 1.9** (Non-Malleable Extractor). A function $\mathrm{nmExt} : [R]^{2n} \to \{0,1\}^m$ is a non-malleable extractor for min-entropy k and error ϵ if the following holds: If X is a source (on $[R]^n$) with min-entropy k, and $f : [R]^n \to [R]^n$ is any function with no fixed points, then

$$|\mathrm{nmExt}(X \circ U_{[R]^n}) \circ \mathrm{nmExt}(X \circ f(U_{[R]^n})) \circ U_{[R]^n} - U_m \circ \mathrm{nmExt}(X \circ f(U_{[R]^n}) \circ U_{[R]^n}| \leq \epsilon\,.$$

The first explicit construction of a non-malleable extractors was given in [13], with subsequent improvements of parameters achieved in [12, 29]. However these constructions require min-entropy $> 0.49n$. In a recent work [7], the min-entropy required was improved to $O(\log^2 n)$.

We initiate the study of non-malleable extractors in the interleaved model, where the extractor is guaranteed to work even when symbols from the source X and tampered seed $U_{[R]^n}$ arrive to the non-malleable extractor in a fixed but unknown interleaved order.

We formally define interleaved-non-malleable extractors.

▶ **Definition 1.10** (Interleaved-Non-Malleable Extractor). A function $\mathrm{nmExt} : [R]^{2n} \to \{0,1\}^m$ is a non-malleable extractor in the any-order model for min-entropy k and error ϵ if the following holds: If X is a source (on $[R]^n$) with min-entropy k, $f : [R]^n \to [R]^n$ is any function with no fixed points and $t : [2n] \to [2n]$ is any permutation, then

$$|\mathrm{nmExt}((X \circ U_{[R]^n})_t) \circ \mathrm{nmExt}((X \circ f(U_{[R]^n}))_t) \circ U_{[R]^n} - U_m \circ \mathrm{nmExt}((X \circ f(U_{[R]^n}))_t) \circ U_{[R]^n}| \leq \epsilon\,,$$

where U_m is independent of $U_{[R]^n}$.

In the above definition, when the seed has some min-entropy instead of being uniform, we say that the interleaved-non-malleable extractor is weak-seeded.

Our Results

We give the first explicit construction of an interleaved-non-malleable extractor. Further our non-malleable extractor is weak-seeded.

▶ **Theorem 1.11.** *There exists $\lambda > 0$ such that for any $\delta > 0$, $c > c(\delta)$ and any prime $p > 2^{\frac{\lambda}{\delta}}$, there exists an explicit function $\mathrm{nmExt} : \mathbb{F}_p^{2n} \to \{0,1\}^m$, $m = O(\log n)$, such that if X, Y are independent sources on $\mathbb{F}_p^n$ with min-entropy k_1, k_2 respectively, satisfying $k_1 > (\frac{1}{2} + \delta)n \log p$ and $k_2 > cm$, $t : [2n] \to [2n]$ is any permutation and $f : \mathbb{F}_p^n \to \mathbb{F}_p^n$ is any function with no fixed points, then*

$$|\mathrm{nmExt}((X \circ Y)_t) \circ \mathrm{nmExt}((X \circ f(Y))_t) \circ Y - U_m \circ \mathrm{nmExt}((X \circ f(Y))_t) \circ Y| = n^{-\Omega(1)}.$$

As before, if we are allowed to run the non-malleable extractor in sub-exponential time, we can extract $\Omega(n)$ bits with error $2^{-\Omega(n)}$. See Theorem 7.4 for more details.

Organization

We outline our constructions in Section 2. We introduce preliminaries in Section 3 and recall some known explicit constructions and other tools in Section 4. In Section 6, we present our extractor constructions for 2-interleaved sources. In Section 7, we present our constructions of interleaved-non-malleable extractors. We present the proof of Theorem 1.8 in Section 8.

2 Outline of Constructions

2.1 Extractors for 2-Interleaved Sources

Our extractor for interleaved sources exploits the existence of good 2-source extractors which are functions of $X + Y$. To do this, we encode our source in a new way. Our encoding is based on explicit constructions of certain combinatorial sets, which we call spanning vectors.

▶ **Definition 2.1.** A set of vectors $S \subseteq \mathbb{F}_p^{\bar{\ell}}$ is (r, s)-spanning if the span of any r vectors of S has dimension at least s.

Note that this is the same as a subspace-evasive set: Any $(s-1)$-dimensional subspace contains at most $(r-1)$ vectors in the set. However our parameters are quite different than studied previously [19, 16].

Our explicit constructions of spanning vectors are based on using the columns of a parity check matrix of a linear codes with good erasure list-decodability. Informally, an (e, L)-erasure list-decodable code $\mathcal{C}$ satisfies the property that at most L codewords agree on any particular subset of coordinates of size $n - e$. This property can then be used to lower bound the rank of any subset of e columns of the parity check matrix of $\mathcal{C}$. We refer the reader to Section 5 for more details.

We define the following encoding based on spanning vectors.

▶ **Definition 2.2.** For any (r, s)-spanning set $S = \{v_1, \ldots, v_\ell\} \subseteq \mathbb{F}_p^{\bar{\ell}}$ of size ℓ, the function $\mathrm{enc} : \mathbb{F}_p^{\ell} \to \mathbb{F}_p^{\bar{\ell}}$ defined as

$$\mathrm{enc}(z) = \sum_{i=1}^{\ell} z_i v_i$$

is called an (r, s)-encoding from $\mathbb{F}_p^{\ell}$ to $\mathbb{F}_p^{\bar{\ell}}$.

Consider the following setting: Let $Z = (X \circ Y)_t$ be any 2-interleaved source on $\{0,1\}^{2n}$, where X and Y are arbitrary independent sources on $\{0,1\}^n$ with min-entropy k_1 and k_2 respectively, and $t : [2n] \to [2n]$ is any permutation.

Our first step is to use an (n, s)-encoding enc from $\mathbb{F}_2^{2n}$ to $\mathbb{F}_2^{\bar{n}}$ to encode Z. Thus,

$$\mathrm{enc}(Z) = X' + Y' ,$$

where

$$X' = \sum_{i=1}^{n} X_i v_{t(i)} , \qquad Y' = \sum_{i=j}^{n} Y_j v_{t(n+j)} ,$$

where $S = \{v_1, \ldots, v_{2n}\}$ is an (n, s)-spanning set of vectors.

The idea is to argue that the independent sources X' and Y' (on $\{0,1\}^{\bar{n}}$) have enough min-entropy. Since (by construction) the span of the set of vectors $\{v_{t(1)}, \ldots, v_{t(n)}\}$ has dimension at least s, Lemma 4.9 implies that $H_\infty(X') = k_1' \geq k_1 - (n - s)$. Similarly $H_\infty(Y') = k_2' \geq k_2 - (n - s)$.

We now associate $\mathbb{F}_2^{\bar{n}}$ with $\mathbb{F}_{2^{\bar{n}}}$. A character sum estimate of Karatsuba[1] [24, 25] implies that for any nonprincipal multiplicative character χ of $\mathbb{F}_{2^{\bar{n}}}^*$,

$$\mathbb{E}_{X'} |\mathbb{E}_{Y'}[\chi(X' + Y')]| \leq 2^{-\delta k_2'}$$

[1] This character sum was also used in [10] for constructing explicit two-source extractors.

whenever: $k_1 \geq (\frac{1}{2} + 3\delta)\bar{n} + (n - s)$ and $k_2 \geq \frac{4}{\delta} \log \bar{n} \log p + (n - s)$.

Suppose k_1 and k_2 satisfy these conditions.

We then follow a standard approach and define the function:

$$\text{ext}(Z) = \log_g(X' + Y') \pmod{M},$$

where $M = 2^{\delta k_2'/2}$ and g is a primitive element of $\mathbb{F}_{2^{\bar{n}}}$. Using a version of the Abelian XOR lemma (see Lemma 4.5), it follows that ext is an extractor with output length $\delta k_2'/2$ and error $2^{-\Omega(k_2')}$. Further the extractor is strong in the source X. However, the running time of this extractor is subexponential since it involves computing discrete logs over finite fields. This gives us a semi-explicit extractor construction.

To get a polynomial time extractor, we compute discrete log over a smaller multiplicative subgroup of $\mathbb{F}^*_{2^{\bar{n}}}$. Let $M | 2^{\bar{n}} - 1$ and $M = n^\lambda$ for any constant λ (we show in Theorem 6.2 that we can ensure that there is always such an M). Define the function:

$$\text{ext}_1(Z) = \text{enc}(Z)^{\frac{2^{\bar{n}}-1}{M}}.$$

Thus $\text{ext}_1(Z)$ is a distribution on the multiplicative subgroup $G = \{x^{\frac{2^{\bar{n}}-1}{M}} : x \in \mathbb{F}^*_{2^{\bar{n}}}\}$ (of $\mathbb{F}^*_{2^{\bar{n}}}$) of size M (in fact $\text{ext}_1(Z)$ is a distribution on $G \cup \{0\}$, but $\Pr[\text{ext}_1(Z) = 0] = 2^{-\Omega(n)}$ and hence we ignore this and add this to the error). Let g be a generator of G. It now follows by using the character sum estimate of Karatsuba [24] that the function:

$$\text{ext}(Z) = \log_g(\text{ext}_1(Z))$$

is an extractor.

We need to find a generator g of G efficiently. For this, we use an efficient algorithm of Shoup [39] for finding a small set of elements such that one of them is a primitive element of $\mathbb{F}_{2^{\bar{n}}}$. We use a straightforward method to find g from this set in polynomial time. We achieve output length of $\lambda \log n$ and error $n^{-\Omega(1)}$. The extractor is strong in the source X.

Reducing the Min-Entropy Rate. For some c and any $\delta > 0$, let $p > 2^{\frac{c}{\delta}}$ be any prime. When the source $Z = (X \circ Y)_t$ is on $[p]^{2n}$, we can reduce the min-entropy rate requirement of the source X to $(\frac{1}{2} + \delta)$. The construction follows the same outline as above (using (n, s)-encodings from $\mathbb{F}_p^{2n}$ to $\mathbb{F}_p^{\bar{n}}$), and the improvement is achieved by using the fact that over alphabet $[p]$, we can construct (n, n)-spanning sets in $\mathbb{F}_p^{\bar{n}}$ with $\bar{n} = n(1 + \frac{\delta}{5})$ (using explicit codes from [20]). The output length of the extractor obtained is $\lambda \log n$ (for any constant λ) and achieves error $n^{-\Omega(1)}$. Further the extractor is strong in the source X.

Improving the Output Length. We improve the output length of the above extractor to $\Omega(n)$ when both sources X and Y (on $[p]^n$) have min-entropy at least $(\frac{1}{2} + \delta)n \log p$. Our construction is as follows. Let SExt be an explicit strong seeded extractor for linear min-entropy with linear output length and polynomially small error with seed seed length $O(\log n)$, for example from the work of [21]. Let $Z_{[n]}$ denote the projection of Z to the first n coordinates and let ext_p denote the extractor constructed in the previous paragraph (for 2-interleaved sources on $[p]^{2n}$). Our extractor is the following function:

$$\text{ext}_{p,long}(Z) = \text{SExt}(Z_{[n]}, \text{ext}_p(Z)).$$

We sketch the proof of correctness. Without loss of generality, suppose that X has more symbols in $Z_{[n]}$ than the source Y. Let $S \subseteq [n]$ be the coordinates of X which are in $Z_{[n]}$

and let X_S denote the projection of X to the coordinates indexed by S. Let $T \subset [n]$ be the coordinates of Y which are in $Z_{[n]}$ and let Y_T denote the projection of Y to the coordinates indexed by T. Further, we use $X_S \circ Y_T$ to denote $Z_{[n]}$. Note that, by assumption $|S| \geq \frac{n}{2}$ and $|T| \leq \frac{n}{2}$. It follows by Lemma 4.7 that $Y|Y_T$ is close to a source with min-entropy $> \frac{\delta n \log p}{2}$ with probability $1 - 2^{-\Omega(n)}$. Also note that X_S has min-entropy $\geq \delta n \log p$.

Consider such a good fixing $Y_T = y_T$. Since X and $Y|Y_T = y_T$ have enough min-entropy, it follows that even under this fixing, $W = \text{ext}_p(Z)$ is close to uniform. We now use the property that ext_p is strong with respect to the source X_S, i.e.,

$$|(X_S, W) - (X_S, U_d)| \leq n^{-\Omega(1)} .$$

Using a probability lemma from [38], it follows that for any $W = w$,

$$|X_S - (X_S|(W = w))| \leq n^{-\Omega(1)} ,$$

(using that w is of length $O(\log n)$).

Hence, $\text{SExt}(X_S \circ Y_T, W)|Y_T = y_T$ is $n^{-\Omega(1)}$-close to the convex combination: $\sum_w \Pr[(W|Y_T = y_T) = w]\text{SExt}(X_S \circ Y_T, w)|Y_T = y_T$. Since as observed above, $W|Y_T = y_T$ is $n^{-\Omega(1)}$-close to U_d, it follows that $\text{SExt}(X_S \circ Y_T, W)|Y_T = y_T$ is $n^{-\Omega(1)}$-close to $\text{SExt}(X_S \circ y_T, U_d)$. The correctness now follows using the fact that SExt is a seeded extractor for linear min-entropy.

Probabilistic Method. We show in Lemma 5.10, that a random set $S \subset \mathbb{F}_2^n$ of size $2n$ is an $(n, n - 2\sqrt{n})$-spanning set with high probability. Thus, using the proof technique described above, any explicit construction of such a set will yield explicit extractors for 2-interleaved sources on $\{01\}^{2n}$ when one source has min-entropy at least $0.51n$ and the other source has min-entropy at least $cn^{\frac{1}{2}}$. We leave it as an interesting open problem to explicitly construct such a set S.[2]

We give formal proofs of the above extractor constructions and other related constructions in Section 6.

2.2 Interleaved-Non-Malleable Extractors

For some $c > 0$ and any $\delta > 0$, let $p > 2^{\frac{c}{\delta}}$ be any prime. Let X be a source on $[p]^n$ with min-entropy k_1 and Y be a weak- eed on $[p]^n$ with min-entropy k_2. Let $f : [p]^n \to [p]^n$ be any function with no fixed points. Thus the non-malleable extractor has access to $Z = (X \circ Y)_t$ for an artitrary permutation $t : [2n] \to [2n]$. Let Z_f denote the tampered source $(X \circ f(Y))_t$.

We show that the extractor ext_p constructed for 2-interleaved sources (described in the previous section) is also non-malleable. We prove it in the following way. Recall the construction of ext_p:

$$\text{enc}(Z) = \sum_{i=1}^{2n} Z_i v_i , \qquad \text{ext}_1(Z) = \text{enc}(Z)^{\frac{p^{\bar{n}}-1}{M}} , \qquad \text{ext}_p(Z) = \log_g(\text{ext}_1(Z)) ,$$

where $S = \{v_1, \ldots, v_{2n}\}$ is an (n, n)-spanning set in $\mathbb{F}_p^{\bar{n}}$, $M = \text{poly}(n)$, $\bar{n} = n(1 + \frac{\delta}{5})$ and g is a generator of the multiplicative subgroup $G = \{x^{\frac{2^{\bar{n}}-1}{M}} : x \in \mathbb{F}_{2^{\bar{n}}}^*\}$.

[2] This is related to finding explicit constructions of binary erasure list-decodable codes with almost optimal parameters. See Section 5 for more details.

Since ext_p is a distribution on $\mathbb{Z}_M$, it follows by a version of the Abelian XOR lemma proved in [13] that to prove non-malleability, it is enough to prove the bound:

$$|\mathbb{E}[\psi_a(\text{ext}_p(Z))\psi_b(\text{ext}_p(Z_f))]| \leq n^{-\Omega(1)},$$

for all additive characters ψ_a and ψ_b (of $\mathbb{Z}_M$) such that ψ_a is nontrivial. When ψ_b is the trivial character, the above quantity can be bounded by the fact that ext_p is an extractor for 2-interleaved sources. Thus, suppose both ψ_a and ψ_b are nontrivial.

It follows that

$$|\mathbb{E}[\psi_a(\text{ext}_p(Z))\psi_b(\text{ext}_p(Z_f))]| = |\mathbb{E}[\chi_a(\text{enc}(Z))\chi_b(\text{enc}(Z_f))]|$$

where χ_a and χ_b are nonprincipal multiplicative characters of $\mathbb{F}_{2^{\bar{n}}}^*$.

Further, $Z = \sum_{i=1}^n X_i v_{t(i)} + \sum_{j=1}^n Y_j v_{t(j)}$ and $Z_f = \sum_{i=1}^n X_i v_{t(i)} + \sum_{j=1}^n f(Y)_j v_{t(j)}$. Thus,

$$Z = X' + Y', \qquad Z_f = X' + f'(Y'),$$

where $X' = \sum_{i=1}^n X_i v_{t(i)}$, $Y' = \sum_{i=j}^n Y_j v_{t(n+j)}$ and $f' = L \circ f \circ L^{-1}$, L being the one-one linear map $L(z) = \sum_{i=1}^n z_i v_{t(n+i)}$. Thus,

$$|\mathbb{E}[\psi_a(\text{ext}_p(Z))\psi_b(\text{ext}_p(Z_f))]| = |\mathbb{E}[\chi_a(X' + Y')\chi_b(X' + f'(Y'))]|.$$

Using the work of Dodis et al. [13], we can prove the required upper bound on the quantity on the right hand side if f' does not have any fixed points. We indeed show that f' has no fixed points (by using the fact that L is one-one and f has no fixed points). This completes the proof sketch. The non-malleable extractor outputs $\lambda \log n$ bits (for any constant λ) and achieves error $n^{-\Omega(1)}$.

See Section 7 for more details.

3 Preliminaries

3.1 Notation

We use capital letters to denote distributions and their support. We use corresponding small letters to denote a sample from the source.

We use $[l]$ to denote the set $\{1, 2, \ldots, l\}$ and $[a, b]$ to denote the set $\{a, a+1 \ldots, b\}$.

We use U_m to denote the uniform distribution over $\{0,1\}^m$.

For any set S, let U_S denote the uniform distribution on S. Also let $s \sim S$ denote a uniform draw from S.

For any string $s \in [R]^n$ and $i \in [n]$, let s_i denote the symbol at the ith coordinate of s. For any one-one map $t : [n] \to [n]$, define the string $w = (s)_t \in [R]^n$ such that $w_i = s_{t(i)}$ for $i = 1, \ldots, n$. Further for any $t \subset [n]$, let s_T denote the $|T|$ length string that is the projection of s onto the coordinates indexed by T.

For any $x \in [p]^{n_1}$, $y \in [p]^{n_2}$ and disjoint subsets $S, T \subset [n_1 + n_2]$ with $|S| = n_1$, $|T| = n_2$, we define $z = x_S \circ y_T$ such that $z_S = x$ and $z_T = y$.

For any integer $M > 0$, let $e_M(x) = e^{\frac{2\pi i x}{M}}$.

3.2 Min-Entropy and Flat Distributions

▶ **Definition 3.1.** The min-entropy of a source X is defined as:

$$H_\infty(X) = \min_{s \in \text{support}(X)} \left\{ \frac{1}{\log(\Pr[X = s])} \right\}.$$

▶ **Definition 3.2.** A distribution (source) D is flat if it is uniform over a set S.

▶ **Definition 3.3.** A (n, k)-source is a distribution on $\{0,1\}^n$ with min-entropy k.

Any (n, k)-source is a convex combination of flat sources supported on sets of size 2^k [43].

3.3 Statistical distance and Convex Combination of Distributions

▶ **Definition 3.4.** Let D_1 and D_2 be two distributions on a set S. The statistical distance between D_1 and D_2 is defined to be: $|D_1 - D_2| = \frac{1}{2}\sum_{s\in S} |\Pr[D_1 = s] - \Pr[D_2 = s]|$.

A distribution D_1 is ϵ-close to another distribution D_2 if $|D_1 - D_2| \le \epsilon$, denoted $D_1 \approx_\epsilon D_2$.

▶ **Definition 3.5.** For random variables X and Y, we use $X|Y$ to denote a random variable with distribution: $\Pr[(X|Y) = x] = \sum_{y\in \text{support}(Y)} \Pr[Y = y] \cdot \Pr[X = x|Y = y]$.

4 Some Known Explicit Constructions and Other Tools

To construct our extractors, we use a variety of tools. We first set up these tools in this section and present our extractor constructions in the next section.

4.1 A 2-Source Extractor

The following double character sum estimate was obtained by Karatsuba [24, 25].

▶ **Theorem 4.1** ([24, 25])**.** *Let p be any prime. Let χ be a nonprincipal multiplicative character of $\mathbb{F}_{p^n}^*$. For any subsets $A, B \subseteq \mathbb{F}_{p^n}$, the following holds: For any integer $\lambda > 0$,*

$$\sum_{a\in A}\left|\sum_{b\in B}\chi(a+b)\right| \le 2\lambda|A|^{\frac{2\lambda-1}{2\lambda}}(|B|p^{\frac{n}{4\lambda}} + |B|^{\frac{1}{2}}p^{\frac{n}{2\lambda}}).$$

The above theorem can be equivalently restated as a result on 2-source extractors.

▶ **Theorem 4.2.** *Let p be any prime. Let χ be a nonprincipal multiplicative character of $\mathbb{F}_{p^n}^*$. For any $\delta > 0$ and independent sources X, Y on $\mathbb{F}_{p^n}$ with min-entropy k_1, k_2 respectively, satisfying $k_1 \ge \left(\frac{1}{2} + 3\delta\right) n \log p$ and $k_2 \ge (4 \log n \log p)/\delta$, we have*

$$\mathbb{E}_{x\sim X}|\mathbb{E}_{y\sim Y}[\chi(x+y)]| \le 2^{-\delta k_2}.$$

Proof. Let X, Y be flat sources on sets A and B respectively. Thus $|A| = 2^{k_1}$ and $|B| = 2^{k_2}$. Setting $\lambda = \frac{n\log p}{\delta k_2}$ in Theorem 4.1 (so that $|B| = 2^{k_2} = p^{\frac{n}{\lambda}}$), we have

$$\begin{aligned}
\mathbb{E}_{x\sim X}|\mathbb{E}_{y\sim Y}[\chi(x+y)]| &\le 2\lambda|A|^{-\frac{1}{2\lambda}}(p^{\frac{n}{4\lambda}} + |B|^{-\frac{1}{2}}p^{\frac{n}{2\lambda}}) \\
&\le 2\lambda|A|^{-\frac{1}{2\lambda}}(p^{\frac{n}{4\lambda}} + 1) \\
&< 3np^{-\frac{3\delta n}{2\lambda}} \\
&= 2^{\log(3n) - \frac{3k_2\delta n\log p}{2n\log p}} < 2^{-\delta k_2}.
\end{aligned}$$

◀

4.2 A Seeded Extractor

We recall an explicit construction of a strong seeded extractor with optimal parameters.

▶ **Theorem 4.3** ([21]). *There exists a constant $\alpha > 0$ such that for all $n, k \in \mathbb{N}$, there exists an explicit strong seeded extractor* $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$, *where* $d = O(\log(n/epsilon))$ *and* $m = (1-\alpha)k$.

4.3 Abelian XOR Lemmas

The following lemma is known as Vazirani's XOR Lemma.

▶ **Lemma 4.4.** *Let D be a distribution over $\mathbb{Z}_M$ such that for every nontrivial additive character ψ of $\mathbb{Z}_M$, we have $|\mathbb{E}[\psi(D)]| \leq \epsilon$. Then, we have*

$$|D - U_M| \leq \epsilon\sqrt{M}\,.$$

Let $\sigma_M : \mathbb{Z}_N \to \mathbb{Z}_M$ be defined as $\sigma_M(x) = x \pmod M$. The following general version of the above XOR lemma was proved in [34].

▶ **Lemma 4.5** ([34]). *Let D be a distribution over $\mathbb{Z}_N$ such that for every non-trivial additive character ψ of $\mathbb{Z}_N$, we have $|\mathbb{E}[\psi(D)]| \leq \epsilon$. Then, for any $M < N$, we have*

$$|\sigma_M(D) - U_M| \leq O(\epsilon \log N\sqrt{M}) + O(M/N)\,.$$

We also record a more generalized form of the XOR Lemma [13].

▶ **Lemma 4.6** ([13]). *Let D_1, D_2 be distributions over $\mathbb{Z}_N$ such that for arbitrary characters ψ, ϕ of $\mathbb{Z}_N$, we have $|\mathbb{E}[\psi(D_1)\phi(D_2)]| \leq \epsilon$, whenever ψ is nontrivial. Then, for any $M < N$, we have*

$$|(\sigma_M(D_1), \sigma_M(D_2)) - (U_M, \sigma_M(D_2))| = O(\epsilon(\log N)^2 M) + O(M/N)\,.$$

4.4 Probability Lemmas

The following result follows from a lemma proved in [32].

▶ **Lemma 4.7** ([32]). *Let X, Y be random variables with supports $S, T \subseteq V$ such that (X, Y) is ϵ-close to a distribution with min-entropy k. Further suppose that the random variable Y can take at most l values. Then*

$$\Pr_{y\sim Y}\left[(X|Y=y) \text{ is } 2\epsilon^{1/2}\text{-close to a source with min-entropy } k - \log l - \log\left(\frac{1}{\epsilon}\right)\right] \geq 1 - 2\epsilon^{1/2}.$$

We also need the following lemma.

▶ **Lemma 4.8** ([38]). *Let Y be a random variable taking values in $\{0,1\}^d$. Suppose $|(X,Y) - (X,U_d)| \leq \epsilon$. Then for any $y \in \mathrm{support}(Y)$, $|X - (X|Y=y)| \leq 2^{d+1}\epsilon$.*

▶ **Lemma 4.9.** *Let X be a source on $\mathbb{F}_p^n$ with min-entropy k. Let $V = \{v_1, \ldots, v_n\}$ be a collection of vectors such that* $\dim(\mathrm{span}\{V\}) \geq n - A$. *Then $X_V = \sum_i x_i v_i : x \sim X$ is a source with min-entropy $\geq k - A\log p$.*

4.5 Finding Primitive Elements in Finite fields

There is no known deterministic polynomial time algorithm to find any primitive element of a finite field $\mathbb{F}_{p^n}$. However, there are efficient algorithms known for a weaker task, where the algorithm is only required to output a small set of elements with the guarantee that one of the elements is primitive. The following result is due to Shoup [39].

▶ **Theorem 4.10** ([39]). *Let $p > 0$ be any prime. For all $n > 0$, there exists a deterministic procedure which takes as input n, runs in time $\mathrm{poly}(n)$, and outputs a set $S = \{a_1, \ldots, a_l\}$, $l = \mathrm{poly}(n)$, such that S contains a primitive element of $\mathbb{F}_{p^n}$.*

5 Constructing Spanning Vectors

A key ingredient in our extractor construction are explicit constructions of spanning vectors. Recall that a set of vectors $S \subseteq \mathbb{F}_p^{\ell}$ is (r, s)-spanning if the span of any r vectors of S has dimension at least s (see Definition 2.1). Our constructions of spanning vectors are simple and are based on explicit linear codes. Recall that a linear code of block length n, dimension k and distance d over any field $\mathbb{F}$ is a k dimensional subspace over $\mathbb{F}$ with the number of zero coordinates of any vector in this subspace being at most $n - d$. The relative rate of the code is k/n and the relative distance is d/n.

We show that the columns of the parity check matrix of any linear code with good erasure list-decoding radius (defined below) can be used as a spanning set.

▶ **Definition 5.1** (Erasure List-Decoding Radius [17]). We say that a linear code $[n, k, d]$ code $\mathcal{C}$ over a finite field $\mathbb{F}$ is (e, L)-erasure list-decodable if for every for every $r \in \mathbb{F}^{n-e}$ and $T \subseteq [n]$ of size $n - e$, $|\{c \in \mathcal{C} : c_T = r\}| \leq L$.

We now establish a simple connection between erasure list-decodable codes and spanning sets.

▶ **Lemma 5.2.** *Let $\mathcal{C}$ be a linear $[n, k, d]$ code over a finite field $\mathbb{F}$, which is (e, L)-earasure list-decodable. Let H be parity check matrix of $\mathcal{C}$, and let S be the set of columns of H. Then $S \subset \mathbb{F}^{n-k}$ is a (r, s)-spanning set of size n, with $r = e$ and $s = e - \log_{|\mathbb{F}|}(L)$.*

Proof. Since $\mathcal{C}$ is (e, L)-erasure list-decodable, it follows that the size of the null space of any e columns of the parity check matrix H is at most L. By the rank-nullity theorem, it follows that the rank of the sub-matrix of H restricted to these e columns is at least $e - \log_{|\mathbb{F}|}(L)$. Thus by definition, the set of columns of H form a $(e, e - \log_{|\mathbb{F}|}(L))$-spanning set. ◀

The following lemma relates the minimum distance of a code to its erasure list-decoding radius, and can be seen as an analogue of the Johnson bound for erasure list-decoding.

▶ **Lemma 5.3** ([18]). *Let $\mathcal{C}$ be a code with block length n and relative distance δ over an alphabet of size q. Then for any $\epsilon > 0$, $\mathcal{C}$ is a (e, L)-erasure list-decodable code, where $e = \left(\frac{q}{q-1} - \epsilon\right)\delta n$ and $L = \frac{q}{(q-1)\epsilon}$.*

Combining the above results, the following lemma is immediate.

▶ **Lemma 5.4.** *For any $\delta > 0$, let $\mathcal{C}$ be a binary linear code with relative distance $\frac{1}{4} + \delta$, and block length $2n$. Then the columns of the parity check matrix of H form a (r, s)-spanning set, with $r = n$ and $s = n - \log\left(\frac{1}{\delta}\right)$.*

Proof. Using Lemma 5.3, it follows that $\mathcal{C}$ is $(n, \frac{1}{\delta})$-erasure list-decodable. Now applying Lemma 5.2, the lemma follows directly. ◀

A similar result follows for the case of q-ary linear codes.

▶ **Lemma 5.5.** *For any $\delta > 0$, let $\mathcal{C}$ be a linear code with relative distance $\frac{q-1}{2q} + \delta$ and block length $2n$ over a finite field of size q. Then the columns of the parity check matrix of H form a (r, s)-spanning set, with $r = n$ and $s = n - \log\left(\frac{q}{(q-1)\delta}\right)$.*

To instantiate the above results, we recall some explicit code constructions. Using standard code concatenation, there are known constructions of binary linear codes achieving the Zyablov bound.

▶ **Theorem 5.6.** *For any $\epsilon, \gamma > 0$, there exists an explicit construction of a binary linear code with relative distance $\delta = \frac{1}{4} + \epsilon$ and relative rate $R \geq \max_{0<r<1-H(\delta+\epsilon)} r\left(1 - \frac{\delta}{H^{-1}(1-r)-\epsilon}\right)$.*

Over larger alphabets, the following explicit codes were constructed in the work of Guruswami and Indyk [20].

▶ **Theorem 5.7** ([20])**.** *There exists $c > 0$ such that for every $\gamma > 0$ and any prime $p > 2^{\frac{c}{\gamma}}$ there is an efficient construction of a linear code $C \subset \mathbb{F}_p^n$ with relative distance $\delta = \frac{1}{2} - \frac{1}{4p}$ and rate $R = \frac{1}{2} - \gamma$.*

Using the above codes, we now have explicit constructions of spanning sets.

▶ **Lemma 5.8.** *There exist constants $\gamma > 0$ and c such that for any n, there exists an explicit $(n, n-c)$-spanning set $S \subset \mathbb{F}_{2^{\bar{n}}}$ of size $2n$, where $\bar{n} = 2n(1-\gamma)$.*

Proof. Let H be the parity check matrix of the explicit linear code $C \subset \mathbb{F}_2^{2n}$ from Theorem 5.6 for relative distance $\frac{1}{4} + \delta$, for some small constant δ. Let $S = \{v_1, \ldots, v_{2n}\}$ be the set of columns of H. Thus $S \subset \mathbb{F}_2^{\bar{n}}$, $\bar{n} = 2n(1-\gamma)$, γ being the relative rate of the code. Applying Lemma 5.4, the result is now immediate. ◀

▶ **Lemma 5.9.** *There exists $c > 0$ such that for any $\gamma > 0$ and any prime $p > 2^{\frac{c}{\gamma}}$, there is an efficient construction of an explicit set $(n, n-C)$-spanning set $S \subset \mathbb{F}_{2^{\bar{n}}}$ of size $2n$, where $\bar{n} = n(1+2\gamma)$ and $C = \frac{2c}{\gamma}$.*

Proof. Let H be the parity check matrix of the explicit linear code $C \subset \mathbb{F}_p^{2n}$ from Theorem 5.7 with relative distance $\frac{1}{2} - \frac{1}{4p}$ and rate $\frac{1}{2} - \gamma$. Let $S = \{v_1, \ldots, v_{2n}\}$ be the set of columns of H. The result now follows by Lemma 5.5. ◀

We show that random sets are (r, s)-spanning sets with overwhelmingly high probability. Guruswami's existence proof of subspace evasive [19] targets different parameters and does not apply here. This lemma is more related to the existence of good erasure list-decodable codes.

▶ **Lemma 5.10.** *Let S be a random subset of $\mathbb{F}_2^n$ of size $2n$. Then,*

$$\Pr[S \text{ is not a } (n, n - 2\sqrt{n})\text{-spanning set }] \leq 2^{-n}\,.$$

Proof. Let $t > 0$. Consider any subset $R \subset S$, $|R| = n$. By standard arguments, it follows that

$$\Pr[\dim(\mathrm{span}(R)) \leq n - t] \leq \binom{n}{t}(2^{-t})^t \leq \left(\frac{n}{2^t}\right)^t\,.$$

Thus,

$$\Pr[\exists\ R \subset S,\ |R| = n \text{ with } \dim(\mathrm{span}(R)) \leq n - t] \leq \binom{2n}{n}\left(\frac{n}{2^t}\right)^t \leq 2^{2n - t^2 + t\log n}$$

The lemma follows by setting $t = 2\sqrt{n} + 1$. ◀

6 Extractors for 2-Interleaved Sources

6.1 Extractors for 2-Interleaved Sources on $\{0,1\}^{2n}$

Our extractor constructions are based on encoding the interleaved-sources using spanning vectors. Recall that any (r,s)-encoding from $\mathbb{F}_p^\ell \to \mathbb{F}_p^{\bar{\ell}}$ is defined in the following way: For any (r,s)-spanning set $S = \{v_1, \ldots, v_\ell\} \subseteq \mathbb{F}_p^{\bar{n}}$, the function $\mathrm{enc} : \mathbb{F}_p^\ell \to \mathbb{F}_p^{\bar{\ell}}$ defined as

$$\mathrm{enc}(z) = \sum_{i=1}^{n} z_i v_i$$

is an (r,s)-encoding from $\mathbb{F}_p^\ell \to \mathbb{F}_p^{\bar{\ell}}$.

The following is a key lemma in our extractor constructions.

▶ **Lemma 6.1** (Main Lemma). *Fix any $\delta > 0$. Let p be any prime and let $Z = (X \circ Y)_t$ be any 2-interleaved source on $\mathbb{F}_p^{2n}$, where X and Y are independent sources on $\mathbb{F}_p^n$ with min-entropy k_1 and k_2 respectively, and $t : [2n] \to [2n]$ is any permutation. Also suppose χ is any nonprincipal multiplicative character of $\mathbb{F}_{p^{\bar{n}}}^*$ and* enc *is an arbitrary (n,s)-encoding from $\mathbb{F}_p^{2n}$ to $\mathbb{F}_p^{\bar{n}}$. Then,*

$$\mathbb{E}_X |\mathbb{E}_Y[\chi(\mathrm{enc}(Z))]| \leq 2^{-\delta(k_2 - (n-s)\log p)},$$

whenever

- $k_1 \geq (\frac{1}{2} + 3\delta)\bar{n}\log p + (n-s)\log p$, *and*
- $k_2 \geq \frac{4\log \bar{n}\log p}{\delta} + (n-s)\log p$.

Proof. For any $z \in \mathbb{F}_p^{2n}$, let

$$\mathrm{enc}(z) = \sum_{i=1}^{2n} z_i v_i$$

where $S = \{v_1, \ldots, v_{2n}\} \subset \mathbb{F}_p^{\bar{n}}$ is (n,s)-spanning.

We have,

$$\chi(\mathrm{enc}(Z)) = \chi\left(\sum_{i=1}^{2n} Z_i v_i\right) = \chi\left(\sum_{i=1}^{n} X_i v_{t(i)} + \sum_{j=1}^{n} Y_j v_{t(n+j)}\right)$$

Define the following independent sources:

$$X' = \sum_{i=1}^{n} x_i v_{t(i)} : x \sim X\,, \quad Y' = \sum_{j=1}^{n} y_j v_{t(n+j)} : y \sim Y\,.$$

Using Lemma 4.9, it follows that: $k_1' = H_\infty(X') \geq k_1 - (n-s)\log p$ and $k_2' = H_\infty(Y') \geq k_2 - (n-s)\log p$.

Thus, we have

$$\begin{aligned}
\mathbb{E}_X |\mathbb{E}_Y[\chi(\mathrm{enc}(Z))]| &= \mathbb{E}_{x\sim X}\left|\mathbb{E}_{y\sim Y}\left[\chi\left(\sum_{i=1}^{n} x_i v_{t(i)} + \sum_{j=1}^{n} y_j v_{t(n+j)}\right)\right]\right| \\
&= \mathbb{E}_{X'}\left|\mathbb{E}_{Y'}\left[\chi\left(X' + Y'\right)\right]\right| \\
&= 2^{-\delta k_2'}
\end{aligned}$$

where the last inequality follows using Theorem 4.2. ◀

Using the above main lemma, we construct extractors for 2-interleaved sources on $\mathbb{F}_2^{2n}$.

▶ **Theorem 6.2.** *For some $\delta > 0$ and any $\lambda > 0$, there exists an explicit function* ext : $\{0,1\}^{2n} \to [M]$, *$M = n^{\lambda}$, such that if X and Y are independent sources on $\mathbb{F}_2^n$ with min-entropy k_1, k_2 respectively satisfying $k_1 > (1-\delta)n$ and $k_2 > 35 \max\{\log n, \log M\}$, $t : [2n] \to [2n]$ is any permutation, then*

$$|\text{ext}((X \circ Y)_t) \circ X - U_M \circ X| = 2^{-\Omega(k_2)}.$$

Proof. Let H be the parity check matrix of a code $C \subset \mathbb{F}_2^{2n}$ with relative distance $= \frac{1}{4} + \delta_1$ (for some small constant δ_1) and constant rate R, where we fix R as follows. Let R_Z be the rate of the code from Theorem 5.6. Let $\epsilon_1 << R_Z$ be a small constant. We choose R in the interval $[R_Z - \epsilon_1, R_Z]$ such that $\bar{n} = 2n(1-R)$ is divisible by integer m, $m = \lambda \log n$. Since $2R_Z\epsilon_1 n >> m$, we can indeed find such an R. Fix $M = 2^m - 1$. We note that $M | 2^{\bar{n}} - 1$. Set $\delta = \frac{R}{6}$.

Let $S = \{v_1, \ldots, v_{2n}\}$ be the set columns of H. By Lemma 5.8, S is $(n, n-C)$-spanning, for some constant C. We interpret each v_i as being an element in the field $\mathbb{F}_{2^{\bar{n}}}$. Consider the multiplicative subgroup:

$$G = \{x^{\frac{2^{\bar{n}}-1}{M}} : x \in \mathbb{F}_{2^{\bar{n}}}^*\}.$$

A generator g of G can be found efficiently in the following way: Using Theorem 4.10, we can efficiently construct a set $S = \{a_1, \ldots, a_l\}$, $l = \text{poly}(n)$, such that one of the a_i's, say a_j, is a primitive element of $\mathbb{F}_{2^{\bar{n}}}$. Let $S' = \{a_1^{\frac{2^{\bar{n}}-1}{M}}, \ldots, a_l^{\frac{2^{\bar{n}}-1}{M}}\}$. We note that $a_j^{\frac{2^{\bar{n}}-1}{M}} \in S'$ is an element of order M. Thus, it is enough to enumerate over the elements in S' and compute the order of each element. Since the order of any element in S' is bounded by $M = \text{poly}(n)$, the search procedure can be implemented efficiently.

Let $Z = (X \circ Y)_t$. For any $z \in \mathbb{F}_2^{2n}$, define the functions:

$$\text{enc}(z) = \sum_{i=1}^{2n} z_i v_i\,, \qquad \text{ext}_1(z) = (\text{enc}(z))^{\frac{2^{\bar{n}}-1}{M}}\,, \qquad \text{ext}(z) = \log_g(\text{ext}_1(z))\,.$$

We note that ext_1 and ext are efficiently computable functions. Further note that enc is an $(n, n-C)$-encoding from $\mathbb{F}_2^{2n}$ to $\mathbb{F}_2^{\bar{n}}$.

Using the above lemma, we prove the following claim.

▶ **Claim 6.3.** *Let $\psi(x) = e_M(\beta x)$, $\beta \neq 0 \pmod M$, be any nontrivial character of the additive group $\mathbb{Z}_M$.*

Then,

$$\mathbb{E}_X\, |\mathbb{E}_Y[\psi(\text{ext}_2((X \circ Y)_t))]| \leq 2^{-\delta k_2}\,.$$

We note that Theorem 6.2 follows directly from Claim 6.3 by using Lemma 4.4. Thus it is enough to prove Claim 6.3.

Proof of Claim 6.3. We have,

$$\begin{aligned}\psi(\text{ext}(z)) &= e_M(\beta \log_g(\text{ext}_1(z))) \\ &= \chi\left(\text{enc}(z)\right),\end{aligned}$$

where $\chi(x) = e_M(\beta \log_g(x))$ is a nonprincipal multiplicative character of $\mathbb{F}_{2^{\bar{n}}}^*$ of order $\frac{M}{\gcd(M,\beta)}$.

Thus, we have

$$\mathbb{E}_X \left|\mathbb{E}_Y[\psi(\text{ext}_2((X \circ Y)_t))]\right| = \mathbb{E}_{x\sim X} \left|\mathbb{E}_{y\sim Y}\left[\chi\left(\text{enc}(Z)\right)\right]\right| \leq 2^{-\delta k_2},$$

where the inequality follows from Lemma 6.1. ◀

◀

It is direct from the above theorem, that if we insist that the output of the above extractor is a bit string, we have the following result.

▶ **Theorem 6.4** (Theorem 1.2 restated). *For some $\delta > 0$ and any $\lambda > 0$, there exists an explicit function $\text{ext} : \{0,1\}^{2n} \to \{0,1\}^m$, $m = \lambda \log n$, such that if X, Y are independent sources on $\mathbb{F}_2^n$ with min-entropy k_1, k_2 respectively satisfying $k_1 > (1-\delta)n$ and $k_2 > 35\max\{\log n, m\}$, $t : [2n] \to [2n]$ is any permutation, then*

$$|\text{ext}((X \circ Y)_t) \circ X - U_m \circ X| = n^{-\Omega(1)}.$$

6.2 Extracting from 2-Interleaved Sources on $\mathbb{F}_p^{2n}$

If the sources X and Y are on $\mathbb{F}_p^n$ (for some large enough prime p), we can reduce the min-entropy rate requirement of the source X to about $\frac{1}{2}$.

▶ **Theorem 6.5** (Theorem 1.3 restated). *There exists $c > 0$ such that for any $\delta, \lambda > 0$ and any prime $p > 2^{\frac{c}{\delta}}$, there exists an explicit function $\text{ext}_p : \mathbb{F}_p^{2n} \to \{0,1\}^m$, $m = \lambda \log n$, such that if X and Y are independent sources on $\mathbb{F}_p^n$ with min-entropy k_1, k_2 respectively, satisfying $k_1 > (\frac{1}{2} + \delta)n \log p$ and $k_2 > \frac{5}{\delta}\max\{\log n \log p, m\}$, $t : [2n] \to [2n]$ is any injective map, then*

$$|\text{ext}_p((X \circ Y)_t) \circ X - U_m \circ X| = n^{-\Omega(1)}.$$

Proof. Let $S = \{v_1, \ldots, v_{2n}\}$ be an explicit (n,n-C)-spanning set in $\mathbb{F}_p^{\bar{n}}$ from Lemma 5.9. Further, as in the proof of Theorem 6.2, we choose the rate of the code in Lemma 5.9 such that $m|\bar{n}$ and $m = \lambda \log_p n$. Thus we can ensure that $\bar{n} \leq n(1 + \frac{\delta}{5})$.

Let $M = n^\lambda$. For any $z \in \mathbb{F}_p^{2n}$, define the functions:

$$\text{enc}(z) = \sum_{i=1}^{2n} z_i v_i\,, \qquad \text{ext}_1(z) = (\text{enc}(z))^{\frac{p^{\bar{n}}-1}{M}}\,, \qquad \text{ext}(z) = \log_g(\text{ext}_1(z))$$

where g is a generator of $G = \{x^{\frac{p^{\bar{n}}-1}{M}} : x \in \mathbb{F}_{p^{\bar{n}}}^*\}$. The proof now follows using Lemma 6.1 and Lemma 4.4. ◀

6.3 Improving the Output Length

The output length of the extractor in Theorem 6.5 is $\Omega(\log n)$. We improve the output length to $\Omega(n)$ bits when the min-entropy rate of both the sources (on $\mathbb{F}_p^n$) are slightly more than $\frac{1}{2}$.

A general technique to improve the output length extractors was introduced by Shaltiel [38]. In particular, Shaltiel showed that the function:

$$\text{SExt}(X, 2\text{ext}(X,Y)) \circ \text{SExt}(Y, 2\text{ext}(X,Y))$$

is 2-source extractor with longer output length, where 2ext is a 2-source extractor with short output length and SExt is a seeded extractor set to appropriate parameters.

However this does not work in our case since it requires access to the individual sources X and Y. Surprisingly, we show that the construction: $\mathrm{SExt}(((X \circ Y)_t)_{[n]}, 2\mathrm{ext}_p((X \circ Y)_t))$ can be proved to be an extractor.

▶ **Theorem 6.6.** *There exists $c > 0$ such that for any $\delta > 0$ and any prime $p > 2^{\frac{c}{\delta}}$, there exists an explicit function $\mathrm{ext}_{p,long} : \mathbb{F}_p^{2n} \to \{0,1\}^m$, $m = \Omega(n)$, such that if X and Y are independent sources on $\mathbb{F}_p^n$ with min-entropy k_1, k_2 respectively satisfying $k_1 > (\frac{1}{2} + \delta)n \log p$ and $k_2 > (\frac{1}{2} + \delta)n \log p$, $t : [2n] \to [2n]$ is any injective map, then*

$$|\mathrm{ext}_{p,long}((X \circ Y)_t) - U_m| = n^{-\Omega(1)}.$$

Proof. Let SExt be the seeded-extractor from Theorem 4.3 with parameters $\beta = \delta$, $\alpha = \delta/2$ and $\epsilon = n^{-\Omega(1)}$. Let the seed length of SExt with this setting of the parameters be $d = \lambda \log n$. Let $Z = (X \circ Y)_t$. Define

$$\mathrm{ext}_{p,long}(Z) = \mathrm{SExt}(Z_{[n]}, \mathrm{ext}_p(Z)),$$

where ext_p is the extractor from Theorem 6.5 designed to extract from 2-interleaved sources with one source at min-entropy $k_1 \geq (\frac{1}{2} + \delta)n \log p$ and the other source with min-entropy $k_2 \geq \frac{\delta n \log p}{2}$ with error $\epsilon_p = n^{-2\lambda}$ and output length $m_p = \lambda \log n$.

Let $S = \{i \in [n] : Z_i = X_i\}$ and $T = \{j \in [n] : Z_j = Y_j\}$. Also let $\bar{S} = [n] \setminus S$ and $\bar{T} = [n] \setminus T$. Without loss of generality, we can assume that $|S| \geq \frac{n}{2}$. It follows from Lemma 4.7 that there exists a set $Good_y$ such that for any $y_T \in Good_y$, $Y_{\bar{T}}|Y_T = y_T$ is $2^{-\Omega(n)}$-close to a source with entropy more than $\frac{\delta n \log p}{2}$, and $\Pr[Y_t \in Good_y] > 1 - 2^{-\Omega(n)}$.

Let $y_T \in Good_y$. It follows by the setting of ext_p that

$$|(\mathrm{ext}_p(Z|Y_T = y_T) \circ X_S - U_m \circ X_S| \leq n^{-2\lambda}.$$

Using Lemma 4.8, it follows that

$$|X_S - (X_S|(\mathrm{ext}_p(Z|Y_T = y_T) = e))| \leq n^{-\lambda+1}. \tag{1}$$

Let $p_{y_T} = \Pr[Y_T = y_T]$ and let $p_{e|y_T} = \Pr[\mathrm{ext}_p(Z|Y_T = y_T) = e]$.

Using the above estimates, we have

$$\begin{aligned}
|\mathrm{ext}_{p,long}(Z) - U_m| &\leq \sum_{y_T} p_{y_T} |\mathrm{SExt}(X_S \circ y_T, \mathrm{ext}_p(Z|Y_T = y_T)) - U_m| \\
&\leq \left(\sum_{y_T \in Good_y} p_{y_T} |\mathrm{SExt}(X_S \circ y_T, \mathrm{ext}_p(Z|Y_T = y_T)) - U_m| \right) + 2^{-\Omega(n)} \\
&\leq \sum_{y_T \in Good_y} p_{y_T} \left(\sum_e p_{e|y_T} |\mathrm{SExt}(X_S \circ y_T, e) - U_m| + n^{-\lambda+1} \right) + 2^{-\Omega(n)} \\
&\leq \left(\sum_{y_T \in Good_y} p_{y_T} |\mathrm{SExt}(X_S \circ y_T, U_d) - U_m| \right) + n^{-\Omega(1)} \\
&= n^{-\Omega(1)}.
\end{aligned}$$

where the last line follows from the fact that X_S has min-entropy at least $\delta n \log p$. ◀

6.4 One Bit Extractors for 2-Interleaved Sources on $\mathbb{F}_p^{2n}$ with Exponentially Small Error

Note that all our extractor constructions so far have polynomially small error if we insist that the output of the extractor is a bit string. Here we show how to achieve exponentially small error for 2-interleaved sources on $\mathbb{F}_p$, for any large enough prime. However we can output only 1 bit.

▶ **Theorem 6.7.** *There exists $c > 0$ such that for any $\delta > 0$ and any prime $p > 2^{\frac{c}{\delta}}$, there exists an explicit function $\text{ext}_{1bit} : \mathbb{F}_p^{2n} \to \{0,1\}$, such that if X and Y are independent sources on $\mathbb{F}_p^n$ with min-entropy k_1, k_2 respectively, satisfying $k_1 > (\frac{1}{2} + \delta)n \log p$ and $k_2 > (5 \log n \log p)/\delta$, $t : [2n] \to [2n]$ is any injective map, then*

$$|\text{ext}_{1bit}((X \circ Y)_t) \circ X - U_1 \circ X| = 2^{-\Omega(k_2)}.$$

Proof. Let $S = \{v_1, \ldots, v_{2n}\}$ be an explicit (n,n-C)-spanning set in $\mathbb{F}_p^{\bar{n}}$ from Lemma 5.9. Define the functions:

$$\text{enc}(z) = \sum_{i=1}^{2n} z_i v_i \,, \qquad \text{ext}(z) = \text{QR}\left(\text{enc}(z)\right) \,,$$

where QR is the quadratic character of $\mathbb{F}_{p^{\bar{n}}}^*$. The proof now follows using Lemma 6.1. ◀

6.5 Semi-Explicit Extractors for 2-Interleaved Sources with Linear Output Length and Exponentially Small Error

We note that the extractors constructed so far have either achieved linear output length or exponentially small error, but not both simultaneously. We show that if we allow the extractors to run in sub-exponential time, then we can indeed construct such extractors. (Note that the trivial algorithm to find such an extractor runs in doubly exponential time.) The non-polynomial running time comes from having to compute the discrete logarithm. To reduce the running time, we can in fact use a heuristic algorithm for finding discrete logarithm [2], which runs in time $n^{O(\log n)}$ on fields of small characteristics under plausible assumptions.

▶ **Theorem 6.8.** *For some $\delta > 0$, there exists a semi-explicit function $\text{ext} : \{0,1\}^{2n} \to \{0,1\}^m$, such that if X and Y are independent sources on $\mathbb{F}_2^n$ with min-entropy k_1, k_2 respectively satisfying $k_1 > (1-\delta)n$ and $k_2 > \frac{10}{\delta} \max\{\log n, m\}$, $t : [2n] \to [2n]$ is any permutation, then*

$$|\text{ext}((X \circ Y)_t) \circ X - U_m \circ X| = 2^{-\Omega(k_2)}.$$

Proof. Let $S = \{v_1, \ldots, v_{2n}\}$ be an explicit $(n, n-C)$-spanning set in $\mathbb{F}_2^{\bar{n}}$ constructed using Lemma 5.8. Let $m = \frac{\delta k_2}{2}$. For any $z \in \mathbb{F}_p^{2n}$, define the functions:

$$\text{enc}(z) = \sum_{i=1}^{2n} z_i v_i \,, \qquad \text{ext}_1(z) = \log_g(\text{enc}(z)) \,, \qquad \text{ext}(z) = \text{ext}_1(z) \pmod{2^m}$$

where g is a generator of $\mathbb{F}_{2^{\bar{n}}}^*$. The proof now follows using Lemma 6.1 and Lemma 4.5. ◀

Using the $(n, n-C)$-spanning sets from Lemma 5.9 to encode the sources, we obtain the following theorem using Lemma 6.1.

▶ **Theorem 6.9.** *There exists $c > 0$ such that for any $\delta > 0$ and any prime $p > 2^{\frac{c}{\delta}}$, there exists a semi-explicit function* $\text{ext} : \mathbb{F}_p^{2n} \to \{0,1\}^m$*, such that if X, Y are independent sources on $\mathbb{F}_p^n$ with min-entropy k_1, k_2 respectively satisfying $k_1 > (\frac{1}{2} + \delta)n \log p$ and $k_2 > \frac{5}{\delta} \max\{\log n \log p, m\}$, $t : [2n] \to [2n]$ is any permutation, then*

$$|\text{ext}((X \circ Y)_t) \circ X - U_m \circ X| = 2^{-\Omega(k_2)}.$$

6.6 Extractors for 2-Interleaved Sources with Linear Min-Entropy Under the Generalized Paley Graph Conjecture

In this section, we show how to construct extractors for sources with linear min-entropy under the widely believed Generalized Paley Graph Conjecture.

▶ **Generalized Paley Graph Conjecture.** *Let χ be any non-principal multiplicative character of $\mathbb{F}_{p^n}^*$. For any constant $\delta > 0$, and arbitrary subsets $A, B \subseteq \mathbb{F}_{p^n}$ satisfying $|A|, |B| > p^{\delta n}$, we have*

$$\left| \sum_{a \in A, b \in B} \chi(a+b) \right| \leq p^{-\gamma(\delta)n} |A||B| \,.$$

Assuming the above conjecture, we obtain the following improved version of Lemma 6.1.

▶ **Lemma 6.10.** *Assume the Generalized Paley graph Conjecture. Fix any $\delta > 0$ and any prime p. Let $Z = (X \circ Y)_t$ be any 2-interleaved source on $\mathbb{F}_p^{2n}$, where X and Y are independent sources on $\mathbb{F}_p^n$ with min-entropy k_1 and k_2 respectively, and $t : [2n] \to [2n]$ is any permutation. Also suppose χ is any nonprincipal multiplicative character of $\mathbb{F}_{p^{\bar{n}}}^*$ and* enc *is an arbitrary (n, s)-encoding from $\mathbb{F}_p^{2n}$ to $\mathbb{F}_p^{\bar{n}}$. Then, there exists $\gamma = \gamma(\delta)$ such that*

$$\mathbb{E}_X |\mathbb{E}_Y[\chi(\text{enc}(Z))]| \leq p^{-\gamma n} \,,$$

whenever

- *$k_1 \geq \delta \bar{n} \log p + (n - s) \log p$, and*
- *$k_2 \geq \delta \bar{n} \log p + (n - s) \log p$.*

Proof. For any $z \in \mathbb{F}_p^{2n}$, let

$$\text{enc}(z) = \sum_{i=1}^{2n} z_i v_i$$

where $S = \{v_1, \ldots, v_{2n}\} \subset \mathbb{F}_p^{\bar{n}}$ is (n, s)-spanning.

We have,

$$\chi(\text{enc}(Z)) = \chi\left(\sum_{i=1}^{2n} Z_i v_i\right) = \chi\left(\sum_{i=1}^{n} X_i v_{t(i)} + \sum_{j=1}^{n} Y_j v_{t(n+j)}\right)$$

Define the following independent sources:

$$X' = \sum_{i=1}^{n} x_i v_{t(i)} : x \sim X \,, \qquad Y' = \sum_{j=1}^{n} y_j v_{t(n+j)} : y \sim Y.$$

Using Lemma 4.9, it follows that: $H_\infty(X') \geq k_1 - (n - s) \log p$ and $H_\infty(Y') \geq k_2 - (n - s) \log p$.

Thus, we have

$$\begin{aligned}\mathbb{E}_X|\mathbb{E}_Y[\chi(\mathrm{enc}(Z))]| &= \mathbb{E}_{x\sim X}\left|\mathbb{E}_{y\sim Y}\left[\chi\left(\sum_{i=1}^{n} x_i v_{t(i)} + \sum_{j=1}^{n} y_j v_{t(n+j)}\right)\right]\right| \\ &= \mathbb{E}_{X'}\left|\mathbb{E}_{Y'}\left[\chi\left(X'+Y'\right)\right]\right| \\ &\leq p^{-\gamma n}\end{aligned}$$

where the last inequality follows using the Generalized Paley Graph Conjecture. ◀

Using the above lemma, we have the following theorem.

▶ **Theorem 6.11.** *Assume the Generalized Paley Graph Conjecture. For any $\delta, \lambda > 0$, there exists an explicit function $\mathrm{ext}_{conjecture} : \{0,1\}^{2n} \to \{0,1\}^m$, $m = \lambda \log n$, such that if X and Y are independent sources with min-entropy δn each, and $t : [2n] \to [2n]$ is any permutation, then*

$$|\mathrm{ext}_{conjecture}((X \circ Y)_t) - U_m| = n^{-\Omega(1)}\,.$$

Proof. Let $S = \{v_1, \ldots, v_{2n}\}$ be an explicit $(n, n-C)$-spanning set in $\mathbb{F}_p^{\bar{n}}$ constructed using Lemma 5.8. Further, as in the proof of Theorem 6.2, we choose the rate of the code in Lemma 5.9 such that $m|\bar{n}$ and $m = \lambda \log n$. Let $M = n^{\lambda}$. For any $z \in \mathbb{F}_2^{2n}$, define the functions:

$$\mathrm{enc}(z) = \sum_{i=1}^{2n} z_i v_i\,, \qquad \mathrm{ext}_1(z) = (\mathrm{enc}(z))^{\frac{p^{\bar{n}}-1}{M}}\,, \qquad \mathrm{ext}(z) = \log_g(\mathrm{ext}_1(z))$$

where g is a generator of $G = \{x^{\frac{2^{\bar{n}}-1}{M}} : x \in \mathbb{F}_{2^{\bar{n}}}^*\}$. The proof now follows using Lemma 6.10 and Lemma 4.4. ◀

We note that assuming the above conjecture, the output length of the above extractor can be improved to $\Omega(n)$ if both X and Y have min-entropy rate more than $\frac{1}{4}$ by using the proof method of Theorem 6.6.

7 Interleaved-Non-Malleable Extractors

In this section, we show that the proof technique developed in constructing extractors for 2-interleaved sources can be used to construct non-malleable extractors in the interleaved model.

▶ **Theorem 7.1.** *There exists $\lambda_1 > 0$ such that for any $\delta, \lambda_2 > 0$, $c > c(\delta)$ and any prime $p > 2^{\frac{\lambda_1}{\delta}}$, there exists an explicit function $\mathrm{nmExt} : \mathbb{F}_p^{2n} \to \{0,1\}^m$, $m = \lambda_2 \log n$, such that if X, Y are independent sources on $\mathbb{F}_p^n$ with min-entropy k_1, k_2 respectively, satisfying $k_1 > (\frac{1}{2} + \delta) n \log p$ and $k_2 > c \max\{m, \log n\}$, $t : [2n] \to [2n]$ is any injective map and $f : \mathbb{F}_p^n \to \mathbb{F}_p^n$ is any function with no fixed points, then*

$$|\mathrm{nmExt}((X \circ Y)_t) \circ \mathrm{nmExt}((X \circ f(Y))_t) \circ Y - U_m \circ \mathrm{nmExt}((X \circ f(Y))_t) \circ Y| = n^{-\Omega(1)}.$$

To prove the above theorem, we recall a character sum estimate of Dodis et al. [13].

▶ **Theorem 7.2.** *For any $\delta > 0$ and $\eta < \frac{1}{2}$, suppose S and T are non-empty subsets of $\mathbb{F}_q$ satisfying $|S| > q^{\frac{1}{2}+\delta}$ and $|T| > \max\{(\frac{1}{\eta})^{\frac{7}{\delta}}, (\log q)^8\}$. Let $f : \mathbb{F}_q \to \mathbb{F}_q$ be any arbitrary function with no fixed points. For arbitrary multiplicative characters χ_a and χ_b, such that χ_a is nonprincipal, we have*

$$\sum_{y\in T}\left|\sum_{x\in S}\chi_a(x+y)\chi_b(x+f(y))\right| < \eta|S||T|.$$

Proof of Theorem 7.1. We use encoding based on spanning vectors. In particular, let $S = \{v_1, \ldots, v_{2n}\}$ be an explicit $(n, n-C)$-spanning set in $\mathbb{F}_p^{\bar{n}}$ constructed using Lemma 5.9. Further, as in the proof of Theorem 6.2, we choose the rate of the code in Lemma 5.9 such that $m|\bar{n}$ and $m = \lambda_2 \log_p n$. Let $M = n^{\lambda_2}$. For any $z \in \mathbb{F}_p^{2n}$, define the functions:

$$\mathrm{enc}(z) = \sum_{i=1}^{2n} z_i v_i\,, \qquad \mathrm{ext}_1(z) = (\mathrm{enc}(z))^{\frac{p^{\bar{n}}-1}{M}}\,, \qquad \mathrm{ext}(z) = \log_g(\mathrm{ext}_1(z))$$

where g is a generator of $G = \{x^{\frac{p^{\bar{n}}-1}{M}} : x \in \mathbb{F}_{p^{\bar{n}}}^*\}$.

We prove the following claim.

▶ **Claim 7.3.** *Let ψ_a and ψ_b be arbitrary characters of the additive group $\mathbb{Z}_M$ such that ψ_a is nontrivial. Then,*

$$\mathbb{E}_{y\sim Y}\left|\mathbb{E}_{x\sim X}[\psi_a(\mathrm{nmExt}((X\circ Y)_t))\psi_b(\mathrm{nmExt}((X\circ f(Y))_t))]\right| = n^{-\Omega(1)}\,.$$

Before proving this claim, we note that Theorem 7.1 follows directly from Claim 7.3 by using Lemma 4.6.

Proof of Claim 7.3. Let $t([n]) = T_1$ and $t([n+1, 2n]) = T_2$. Since S is (n, n)-spanning, it follows that the set $\{v_i : i \in T_1\}$ consists of linearly independent vectors. Similarly $\{v_j : j \in T_2\}$ is a set of linearly independent vectors.

Let $\psi_a(x) = e_M(ax)$, where $a \neq 0 \pmod M$. Also let $\psi_b(x) = e_M(bx)$. If $b = 0 \pmod M$, the claim follows from Lemma 6.1. Thus suppose $b \neq 0 \pmod M$.

We have,

$$\begin{aligned}\psi_a(\mathrm{nmExt}((X\circ Y)_t) &= e_M(a\log_g(\mathrm{ext}_1((X\circ Y)_t)))\\ &= \chi_a\left(\sum_{i=1}^{n} X_i v_{t(i)} + \sum_{j=1}^{n} Y_j v_{t(n+j)}\right)\\ &= \chi_a\left(X' + Y'\right)\end{aligned}$$

where $\chi_a(x) = e_M(a\log_g(x))$ is a nonprincipal multiplicative character of $\mathbb{F}_{p^{\bar{n}}}^*$ of order $\frac{M}{\gcd(M,a)}$, $X' = \sum_{i=1}^{n} x_i v_{t(i)} : x \sim X$ and $Y' = L(Y)$, $L : \mathbb{F}_p^n \to \mathbb{F}_p^{\bar{n}}$ being the injective linear map:

$$L(y) = \sum_{j=1}^{n} y_j v_{t(n+j)}\,.$$

Further,

$$\begin{aligned}\psi_b(\mathrm{nmExt}((X\circ f(Y))_t) &= e_M(b\log_g(\mathrm{ext}_1((X\circ Y)_t)))\\ &= \chi_b\left(\sum_{i=1}^{n} X_i v_{t(i)} + \sum_{j=1}^{n} f(Y)_j Y_{t(n+j)}\right)\\ &= \chi_b\left(X' + f'(Y')\right)\end{aligned}$$

where $f' = L \circ f \circ L^{-1}$ and $\chi_b(x) = e_M(b \log_g(x))$ is a nonprincipal multiplicative character of $\mathbb{F}^*_{p^{\bar{n}}}$ of order $\frac{M}{\gcd(M,b)}$.

We claim that f' has no fixed points. This can be proved in the following way. Suppose $f'(x) = x$ for some x. This implies that $f(L^{-1}(x)) = L^{-1}(x)$ and hence $f(w) = w$ for $w = L^{-1}(x)$. This contradicts our assumption on f. Thus f' has no fixed points.

It now follows from Theorem 7.2 that

$$\mathbb{E}_{x' \sim X'} |\mathbb{E}_{y' \sim Y'} [\chi_a(x' + y')\chi_b(x' + f'(y'))]| = n^{-\Omega(1)}.$$

◀
◀

If we allow the non-malleable extractor to run in sub-exponential time, then using the proof method of the above theorem, it can be shown that the extractor from Theorem 6.9 is non-malleable. Thus, we have the following result.

▶ **Theorem 7.4.** *There exists $\lambda > 0$ such that for any $\delta > 0$, $c > c(\delta)$ and any prime $p > 2^{\frac{\lambda}{\delta}}$, there exists a semi-explicit function* $\mathrm{nmExt} : \mathbb{F}_p^{2n} \to \{0,1\}^m$, *$m = \Omega(n)$, such that if X, Y are independent sources on $\mathbb{F}_p^n$ with min-entropy k_1, k_2 respectively, satisfying $k_1 > (\frac{1}{2} + \delta)n \log p$ and $k_2 > c \max\{m, \log n\}$, $t : [2n] \to [2n]$ is any permutation and $f : \mathbb{F}_p^n \to \mathbb{F}_p^n$ is any function with no fixed points, then*

$$|\mathrm{nmExt}((X \circ Y)_t) \circ \mathrm{nmExt}((X \circ f(Y))_t) \circ Y - U_m \circ \mathrm{nmExt}((X \circ f(Y))_t) \circ Y| = 2^{-\Omega(k_2)}.$$

We note that under the Generalized Paley Graph Conjecture, we can reduce the min-entropy requirement of the source X in Theorem 7.1 to βn, for any constant $\beta > 0$.

8 Proof of Theorem 1.8

We briefly recall some definitions from communication complexity. We refer the reader to [27] for more background. For convenience, we define boolean functions with range $\{-1, 1\}$ (instead of $\{0, 1\}$).

▶ **Definition 8.1.** Let $f : [p]^{2n} \to \{-1, 1\}$ be any function. Fix any equi-partition of $[2n]$ into subsets S, T. For any rectangle R and probability distribution μ on $[p]^{2n}$, denote

$$\mathrm{Disc}^{\mu,R}_{S,T}(f) = |\Pr_{\mu}[f(x_S, y_T) = 1 \text{ and } (x, y) \in R] - \Pr_{\mu}[f(x_S, y_T) = -1 \text{ and } (x, y) \in R]|.$$

▶ **Definition 8.2.** The discrepancy of $f : [p]^{2n} \to \{-1, 1\}$ with respect to an equi-partition of [2n] into S, T and distribution μ on $[p]^{2n}$ is defined as:

$$\mathrm{Disc}^{\mu}_{S,T}(f) = \left\{ \max_R \left(\mathrm{Disc}^{\mu,R}_{S,T}(f) \right) \right\}.$$

▶ **Definition 8.3.** The maximal-equipartition discrepancy of $f : [p]^{2n} \to \{0, 1\}$ with respect to a distribution μ on $[p]^{2n}$ is defined as:

$$\mathrm{Disc}^{\mu}_{best}(f) = \max_{\substack{S,T:|S|=|T|=n, \\ S \cup T = [2n]}} \left\{ \mathrm{Disc}^{\mu}_{S,T}(f) \right\}.$$

The following theorem provides a method to lower bound randomized best-paritition communication complexity of f using its maximal-equi-partition discrepancy. A proof can be found in [27].

▶ **Theorem 8.4.** *For every function $f : [p]^{2n} \to \{-1, 1\}$, every probability distribution μ on $[p]^{2n}$ and every $\epsilon \geq 0$,*

$$R^{best,\frac{1}{2}-\epsilon}(f) \geq \log\left(\frac{2\epsilon}{\mathrm{Disc}^{\mu}_{best}(f)}\right).$$

We now prove Theorem 1.8.

Proof of Theorem 1.8. We show that the explicit extractor from Theorem 6.7 is the required function. Recall the construction of the extractor.

Let $S = \{v_1, \ldots, v_{2n}\}$ be an explicit (n,n-C)-spanning set in $\mathbb{F}_p^{\bar{n}}$ constructed using Lemma 5.9, $\bar{n} = n(1 + 2\delta)$.

Define the functions:

$$\mathrm{enc}(z) = \sum_{i=1}^{2n} z_i v_i\,, \qquad \mathrm{ext}(z) = \mathrm{QR}\left(\mathrm{enc}(z)\right),$$

where QR is the quadratic character of $\mathbb{F}^*_{p^{\bar{n}}}$.

We claim that the randomized best partition discrepancy of ext with error $\frac{1}{2} - p^{-\gamma n}$ is at least $(\frac{1}{4} - \delta - \gamma)n \log p$.

Let μ be the uniform distribution on $[p]^{2n}$.

▶ **Claim 8.5.** *For any equi-partition of $[2n]$ into disjoint subsets S and T,*

$$\log\left(\frac{1}{\mathrm{Disc}^{\mu}_{S,T}(\mathrm{ext})}\right) \geq \left(\frac{1}{4} - \delta\right) n \log p\,.$$

We note that the proof of Theorem 1.8 is direct from Claim 8.5 by using Theorem 8.4.

Proof of Claim 8.5. Fix any rectangle $R = X \times Y$, for arbitrary subsets $X, Y \subseteq [p]^n$. We have,

$$\mathrm{Disc}^{\mu,R}_{S,T}(ext) = \frac{|X||Y|}{p^{2n}} \left|\mathbb{E}_{x\in X, y\in Y}[QR\left(\mathrm{enc}(x_S \circ y_T)\right)]\right|$$

We note that if $|X| \leq p^{\frac{3n}{4}}$ or $|Y| \leq p^{\frac{3n}{4}}$, the claim follows easily.

Thus suppose $|X|, |Y| > p^{\frac{3n}{4}}$. We abuse notation and also use X, Y to denote the flat distributions supported on the sets X and Y respectively. Define the distribution $Z = (X \circ Y)_\pi$, where $\pi : [2n] \to [2n]$ is a permutation defined in the following way: Let $S = \{s_1, \ldots, s_n\}$ and $T = \{t_1, \ldots, t_n\}$ such that $s_1 \leq \ldots \leq s_n$ and $t_1 \leq \ldots \leq t_n$. For any $i \in [n]$, define $\pi(i) = s_i$ and for any $j \in [n + 1, 2n]$, define $\pi(j) = t_j$ (thus, $\pi([n]) = S$ and $\pi([n + 1, 2n]) = T$).

We note that enc is an (n, n)-encoding from $\mathbb{F}_p^{2n} \to \mathbb{F}_p^{\bar{n}}$. Thus,

$$\mathrm{enc}(Z) = X' + Y'\,,$$

where X' and Y' are independent sources on $\mathbb{F}_p^{\bar{n}}$ with $H_\infty(X') = \log(|X|)$ and $H_\infty(Y') = \log(|Y|)$.

Using Theorem 4.1, with $\lambda = 1$, we have

$$|\mathbb{E}[QR\left(X' + Y'\right)]| \leq 2\left(\left(\frac{p^{\bar{n}}}{|X||Y|}\right)^{\frac{1}{2}} + \left(\frac{p^{\frac{\bar{n}}{2}}}{|X|}\right)^{\frac{1}{2}}\right)$$

Thus,

$$\begin{aligned}\mathrm{Disc}^{\mu,R}_{S,T}(ext) &\leq 2\left(\frac{|X||Y|}{p^{2n}}\right)\left(\left(\frac{p^{\bar{n}}}{|X||Y|}\right)^{\frac{1}{2}}+\left(\frac{p^{\frac{\bar{n}}{2}}}{|X|}\right)^{\frac{1}{2}}\right)\\ &\leq 2\left(\frac{|X|^{\frac{1}{2}}|Y|^{\frac{1}{2}}}{p^{2n-\frac{\bar{n}}{2}}}+\frac{|X|^{\frac{1}{2}}}{p^{n-\frac{\bar{n}}{4}}}\right)\\ &\leq 2(p^{-(n-\frac{\bar{n}}{2})}+p^{-\frac{n}{2}+\frac{\bar{n}}{4}})\end{aligned}$$

Since the above estimate holds for any arbitrary rectangle R, we have

$$\log\left(\frac{1}{\mathrm{Disc}^{\mu}_{S,T}(\mathrm{ext})}\right)\geq\left(\frac{1}{4}-\delta\right)n\log p.$$

◀
◀

Acknowledgements. We thank anonymous referees for helpful comments, especially about erasure list-decodable codes.

References

1 Noga Alon and Wolfgang Maass. Meanders, Ramsey Theory and Lower Bounds for Branching Programs. In *IEEE Symposium on Foundations of Computer Science*, pages 410–417, 1986. doi:10.1109/SFCS.1986.31.

2 Razvan Barbulescu, Pierrick Gaudry, Antoine Joux, and Emmanuel Thomé. A heuristic quasi-polynomial algorithm for discrete logarithm in finite fields of small characteristic. In *Advances in Cryptology – EUROCRYPT 2014 – 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11-15, 2014. Proceedings*, pages 1–16, 2014. doi:10.1007/978-3-642-55220-5_1.

3 Manuel Blum. Independent unbiased coin flips from a correlated biased source finite state markov chain. *Combinatorica*, 6(2):97–108, 1986.

4 J. Bourgain. More on the sum-product phenomenon in prime fields and its applications. *International Journal of Number Theory*, 01(01):1–32, 2005. doi:10.1142/S1793042105000108.

5 J. Bourgain, A. A. Glibichuk, and S. V. Konyagin. Estimates for the number of sums and products and for exponential sums in fields of prime order. *Journal of the London Mathematical Society*, 73:380–398, 4 2006. doi:10.1112/S0024610706022721.

6 Jean Bourgain, Nets Katz, and Terence Tao. A sum-product estimate in finite fields, and applications. *Geometric and Functional Analysis GAFA*, 14(1):27–57, 2004. doi:10.1007/s00039-004-0451-1.

7 Eshan Chattopadhyay, Vipul Goyal, and Xin Li. Non-malleable extractors and codes, with their many tampered extensions. In *STOC*, 2016.

8 Eshan Chattopadhyay and David Zuckerman. Eshan chattopadhyay and xin li. In *STOC*, 2016.

9 Eshan Chattopadhyay and David Zuckerman. Explicit two-source extractors and resilient functions. In *STOC*, 2016.

10 Benny Chor and Oded Goldreich. Unbiased Bits from Sources of Weak Randomness and Probabilistic Communication Complexity. *Siam Journal on Computing*, 17:230–261, 1988. doi:10.1137/0217015.

11 Benny Chor, Oded Goldreich, Johan Hasted, Joel Freidmann, Steven Rudich, and Roman Smolensky. The bit extraction problem or t-resilient functions. In *IEEE Symposium on Foundations of Computer Science*, pages 396–407, 1985. doi:10.1109/SFCS.1985.55.

12 Gil Cohen, Ran Raz, and Gil Segev. Non-malleable extractors with short seeds and applications to privacy amplification. In *IEEE Conference on Computational Complexity*, pages 298–308, 2012. doi:10.1109/CCC.2012.21.

13 Yevgeniy Dodis, Xin Li, Trevor D Wooley, and David Zuckerman. Privacy amplification and nonmalleable extractors via character sums. *SIAM Journal on Computing*, 43(2):800–830, 2014.

14 Yevgeniy Dodis and Daniel Wichs. Non-malleable extractors and symmetric key cryptography from weak secrets. In *STOC*, pages 601–610, 2009. doi:10.1145/1536414.1536496.

15 Zeev Dvir, Swastik Kopparty, Shubhangi Saraf, and Madhu Sudan. Extensions to the method of multiplicities, with applications to Kakeya sets and mergers. In *FOCS*, pages 181–190, 2009. doi:10.1109/FOCS.2009.40.

16 Zeev Dvir and Shachar Lovett. Subspace evasive sets. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 351–358. ACM, 2012.

17 Venkatesan Guruswami. List decoding from erasures: bounds and code constructions. *IEEE Transactions on Information Theory*, 49(11):2826–2833, 2003. doi:10.1109/TIT.2003.815776.

18 Venkatesan Guruswami. *List Decoding of Error-Correcting Codes (Winning Thesis of the 2002 ACM Doctoral Dissertation Competition)*, volume 3282 of *Lecture Notes in Computer Science*. Springer, 2004. doi:10.1007/b104335.

19 Venkatesan Guruswami. Linear-algebraic list decoding of folded Reed-Solomon codes. In *Computational Complexity (CCC), 2011 IEEE 26th Annual Conference on*, pages 77–85. IEEE, 2011.

20 Venkatesan Guruswami and Piotr Indyk. Near-optimal linear-time codes for unique decoding and new list-decodable codes over smaller alphabets. In *Proceedings of the Thiry-fourth Annual ACM Symposium on Theory of Computing*, STOC'02, pages 812–821, New York, NY, USA, 2002. ACM. doi:10.1145/509907.510023.

21 Venkatesan Guruswami, Christopher Umans, and Salil P. Vadhan. Unbalanced expanders and randomness extractors from Parvaresh–Vardy codes. *J. ACM*, 56(4), 2009. doi:10.1145/1538902.1538904.

22 Jesse Kamp, Anup Rao, Salil P. Vadhan, and David Zuckerman. Deterministic extractors for small-space sources. *Journal of Computer and System Sciences*, 77:191–220, 2011. doi:10.1016/j.jcss.2010.06.014.

23 Jesse Kamp and David Zuckerman. Deterministic Extractors for Bit-Fixing Sources and Exposure-Resilient Cryptography. *Siam Journal on Computing*, 36:1231–1247, 2007. doi:10.1137/S0097539705446846.

24 A.A. Karatsuba. On a certain arithmetic sum. *Soviet Math Dokl., 12, 1172-1174*, 1971. URL: https://www.researchgate.net/publication/258358497_On_a_certain_arithmetic_sum.

25 AA Karatsuba. The distribution of values of dirichlet characters on additive sequences. In *Doklady Acad. Sci. USSR*, volume 319, pages 543–545, 1991.

26 Sergei Konyagin. A sum-product estimate in fields of prime order. *CoRR*, arXiv:math/0304217, 2003. URL: http://arxiv.org/abs/math/0304217v1.

27 Eyal Kushilevitz and Noam Nisan. *Communication complexity*. Cambridge University Press, 1997.

28 Thomas Lengauer. *Handbook of Theoretical Computer Science (Vol. A)*. MIT Press, Cambridge, MA, USA, 1990. URL: http://dl.acm.org/citation.cfm?id=114872.114888.

29 Xin Li. Non-malleable extractors, two-source extractors and privacy amplification. In *FOCS*, pages 688–697, 2012. doi:10.1109/FOCS.2012.26.

30 Xin Li. Improved constructions of two-source extractors. *Electronic Colloquium on Computational Complexity (ECCC)*, 2015. URL: http://eccc.hpi-web.de/report/2015/125.

31 Chi-Jen Lu, Omer Reingold, Salil P. Vadhan, and Avi Wigderson. Extractors: optimal up to constant factors. In *STOC*, pages 602–611, 2003. doi:10.1145/780542.780630.

32 Ueli M. Maurer and Stefan Wolf. Privacy amplification secure against active adversaries. In *CRYPTO*, pages 307–321, 1997. doi:10.1007/BFb0052244.

33 Noam Nisan and David Zuckerman. Randomness is linear in space. *Journal of Computer and System Sciences*, 52(1):43–52, 1996.

34 Anup Rao. An exposition of Bourgain's 2-source extractor. *Electronic Colloquium on Computational Complexity (ECCC)*, 14(034), 2007.

35 Ran Raz. Extractors with weak random seeds. In *ACM Symposium on Theory of Computing*, pages 11–20, 2005. doi:10.1145/1060590.1060593.

36 Ran Raz and Amir Yehudayoff. Multilinear formulas, maximal-partition discrepancy and mixed-sources extractors. *Journal of Computer and System Sciences*, 77:167–190, 2011. doi:10.1016/j.jcss.2010.06.013.

37 Miklos Santha and Umesh V. Vazirani. Generating quasi-random sequences from semi-random sources. *Journal of Computer and System Sciences*, 33:75–87, 1986. doi:10.1016/0022-0000(86)90044-9.

38 Ronen Shaltiel. How to get more mileage from randomness extractors. In *21st Annual IEEE Conference on Computational Complexity (CCC 2006), 16-20 July 2006, Prague, Czech Republic*, pages 46–60, 2006. doi:10.1109/CCC.2006.24.

39 Victor Shoup. Searching for primitive roots in finite fields. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, May 13-17, 1990, Baltimore, Maryland, USA*, pages 546–554, 1990. doi:10.1145/100216.100293.

40 Luca Trevisan and Salil P. Vadhan. Extracting Randomness from Samplable Distributions. In *IEEE Symposium on Foundations of Computer Science*, pages 32–42, 2000. doi:10.1109/SFCS.2000.892063.

41 J. von Neumann. Various techniques used in connection with random digits. *Applied Math Series*, 12:36–38, 1951. Notes by G.E. Forsythe, National Bureau of Standards. Reprinted in *Von Neumann's Collected Works*, 5:768-770, 1963.

42 Andrew Chi-Chih Yao. Some complexity questions related to distributive computing. In *ACM Symposium on Theory of Computing*, pages 209–213, 1979. doi:10.1145/800135.804414.

43 David Zuckerman. Randomness-optimal oblivious sampling. *Random Struct. Algorithms*, 11(4):345–367, 1997. doi:10.1002/(SICI)1098-2418(199712)11:4<345::AID-RSA4>3.0.CO;2-Z.

Non-Malleable Extractors – New Tools and Improved Constructions

Gil Cohen

California Institute of Technology, Pasadena, USA
coheng@gmail.com

Abstract

A non-malleable extractor is a seeded extractor with a very strong guarantee – the output of a non-malleable extractor obtained using a typical seed is close to uniform even conditioned on the output obtained using any other seed. The first contribution of this paper consists of two new and improved constructions of non-malleable extractors:

- We construct a non-malleable extractor with seed-length $O(\log n \cdot \log\log n)$ that works for entropy $\Omega(\log n)$. This improves upon a recent exciting construction by Chattopadhyay, Goyal, and Li (STOC'16) that has seed length $O(\log^2 n)$ and requires entropy $\Omega(\log^2 n)$.
- Secondly, we construct a non-malleable extractor with optimal seed length $O(\log n)$ for entropy $n/\text{polylog}\, n$. Prior to this construction, non-malleable extractors with a logarithmic seed length, due to Li (FOCS'12), required entropy $0.49n$. Even non-malleable condensers with seed length $O(\log n)$, by Li (STOC'12), could only support linear entropy.

We further devise several tools for enhancing a given non-malleable extractor in a black-box manner. One such tool is an algorithm that reduces the entropy requirement of a non-malleable extractor at the expense of a slightly longer seed. A second algorithm increases the output length of a non-malleable extractor from constant to linear in the entropy of the source. We also devise an algorithm that transforms a non-malleable extractor to the so-called t-non-malleable extractor for any desired t. Besides being useful building blocks for our constructions, we consider these modular tools to be of independent interest.

1998 ACM Subject Classification F.0 Theory of Computation] General, G.3 Probability and Statistics

Keywords and phrases extractors, non-malleable, explicit constructions

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.8

1 Introduction

A *non-malleable extractor* is a seeded extractor with a very strong property – the output of a non-malleable extractor obtained using a typical seed is close to uniform even given the output obtained using any other seed. Constructing non-malleable extractors gained a significant attention in the literature, with original motivation coming from privacy amplification protocols due to Dodis and Wichs [12]. Recently, non-malleable extractors were used as a key component in the breakthrough construction of two-source extractors by Chattopadhyay and Zuckerman [5]. Before giving the formal definition of a non-malleable extractor, we recall the more basic notion of seeded extractors (see [30, 32] for a more elaborated discussion).

Seeded extractors, introduced by Nisan and Zuckerman [26], are central objects in pseudorandomness with many applications in theoretical computer science. Informally speaking, a seeded extractor is a randomized algorithm that uses only few bits of internal randomness, called the seed, to extract pure randomness from a weak random source.

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 8; pp. 8:1–8:29


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

For a formal treatment, we recall the notion of min-entropy, introduced by Chor and Goldreich [6]. A random variable X has min-entropy k if no point is sampled by X with probability larger than 2^{-k}. When X is supported on n bit strings, we say that X is an (n,k)-source. With this notion of entropy, we recall the definition of a seeded extractor.

▶ **Definition 1.1** (Seeded extractors)**.** A function $\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is called a *seeded extractor* for entropy k if for any (n,k)-source X and an independent random variable Y that is uniformly distributed over $\{0,1\}^d$, it holds that $\mathsf{Ext}(X,Y) \approx U_m$.

In the definition above, and throughout the paper, U_m stands for the uniform distribution over m bit strings. Further, by writing $A \approx B$ we mean that A and B are two distributions that are close in statistical distance. Throughout the introduction we will be vague about how close distributions are exactly, and the reader is advised to think of A, B as being, say, $1/10$-close. In some cases, constants that appear in the results described in this section hide $\mathrm{polylog}(1/\varepsilon)$ factors, where ε is the error guarantee.

The second input to Ext is called the *seed*. The general goal is to design efficiently computable seeded extractors with short seeds for low entropy sources, having many output bits. By a straightforward application of the probabilistic method one can prove the existence of a seeded extractor that works for any entropy $k = \Omega(1)$ with seed length $d = \log(n) + O(1)$, and $m = k - O(1)$ output bits. By now, following a long line of research initiated by [26] and that has accumulated to [15, 13, 31], it is known how to construct seeded extractors with seed length $O(\log n)$ for any entropy $k = \Omega(1)$, with $m = 0.99k$ output bits.

For many applications, it is desired that the output of a seeded extractor will be close to uniform even given the seed that was used for the extraction. A seeded extractor that has this property is called *strong*.

▶ **Definition 1.2** (Strong seeded extractors)**.** A function $\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is called a *strong seeded extractor* for entropy k if for any (n,k)-source X and an independent random variable Y that is uniform over $\{0,1\}^d$, it holds that $(\mathsf{Ext}(X,Y),Y) \approx (U_m,Y)$.

In the definition above, U_m stands for a random variable that is uniformly distributed over m bit strings and is independent of Y, namely, (U_m,Y) is a product distribution. The explicit constructions mentioned above [15, 13, 31] are in fact strong. In particular, it is known how to construct a strong seeded extractor for any entropy $k = \Omega(1)$ with seed length $d = O(\log n)$ and $m = 0.99k$ output bits. Moreover, there is a black-box transformation that produces a strong seeded extractor given a seeded extractor (which not necessarily strong) with essentially the same parameters [29].

1.1 Non-malleable extractors

It is straightforward to show that if $\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is a strong seeded extractor for entropy k then for any (n,k)-source X, there exists a small subset of seeds $B \subset \{0,1\}^d$ such that for any $y \notin B$, it holds that $\mathsf{Ext}(X,y)$ is close to uniform. That is, one can associate with any source X a small set of "bad" seeds such that for any seed y that is not bad, $\mathsf{Ext}(X,y)$ is close to uniform.

This dichotomic point of view on strong seeded extractors is frequently used in the literature. Taking this view, we note that nothing in the definition of a strong seeded extractor prevents $\mathsf{Ext}(X,y)$ from being arbitrarily correlated with $\mathsf{Ext}(X,y')$ for some good seeds y, y'. Namely, there is no guarantee on the correlation (or the lack of) between the outputs of a strong seeded extractor when applied with two distinct good seeds. One can then contemplate an even stronger notion of seeded extractors in which the output $\mathsf{Ext}(X,y)$

Table 1 Summary of explicit non-malleable extractors from the literature as well as our contribution.

Construction	Seed length	Supported min-entropy
[12] (non-constructive)	$\log(n) + O(1)$	$\Omega(\log \log n)$
[25]	n	$(0.5 + \delta) \cdot n$ for any constant $\delta > 0$
[9, 10, 17]	$O(\log n)$	$(0.5 + \delta) \cdot n$ for any constant $\delta > 0$
[19]	$O(\log n)$	$(0.5 - \alpha) \cdot n$ for some small constant $0 < \alpha < 0.5$
[4]	$O(\log^2 n)$	$\Omega(\log^2 n)$
Theorem 2.1	$O(\log n \cdot \log \log n)$	$\Omega(\log n)$
Theorem 2.2	$O(\log n)$	$\Omega(n/\log^c n)$ for any constant $c > 0$

is uniform even conditioned $\mathsf{Ext}(X, y')$ for any good seed y and for any $y' \neq y$. This point of view leads to the definition of non-malleable extractors. We choose to present next an equivalent definition, which is the one originally suggested by Dodis and Wichs [12]. In Lemma 4.14 we show that the original definition and the dichotomic one described above are equivalent. On top being natural, we make frequent use of the dichotomic definition in our proofs.

▶ **Definition 1.3** (Non-malleable extractors)**.** A function $\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is called a *non-malleable extractor* for entropy k if for any (n,k)-source X and a function $\mathcal{A}\colon \{0,1\}^d \to \{0,1\}^d$ with no fixed points, it holds that

$$(\mathsf{Ext}(X,Y), \mathsf{Ext}(X,\mathcal{A}(Y)), Y) \approx (U_m, \mathsf{Ext}(X,\mathcal{A}(Y)), Y),$$

where Y is uniformly distributed over $\{0,1\}^d$ and is independent of X.

As suggested in [9], one can consider the generalization to t-non-malleable extractors in which $\mathsf{Ext}(X,y)$ is close to uniform even conditioned on $\mathsf{Ext}(X,y^1), \ldots, \mathsf{Ext}(X,y^t)$ for any good seed y and arbitrary seeds $y^1, \ldots, y^t \in \{0,1\}^d \setminus \{y\}$, or equivalently, where $\mathsf{Ext}(X,Y)$ looks uniform even given $\mathsf{Ext}(X,\mathcal{A}_1(Y)), \ldots, \mathsf{Ext}(X,\mathcal{A}_t(Y))$ for arbitrary functions $\{\mathcal{A}_i\colon \{0,1\}^d \to \{0,1\}^d\}_{i=1}^t$ with no fixed points. Note that a strong seeded extractor can be viewed as a 0-non-malleable extractor. Although this generalization is useful for some applications (e.g., [5] uses $t = \text{polylog}\, n$), in this section we consider only the standard definition of non-malleable extractors, namely, the case $t = 1$. In fact, one of our contributions is an algorithm that transforms a "standard" non-malleable extractor (namely, a 1-non-malleable extractor) to a t-non-malleable extractor, for any desired $t > 1$, in a black-box manner (see Lemma 2.5). Thus, it is not only for simplicity that the reader can focus on standard non-malleable extractors.

Dodis and Wichs [12], who introduced the notion of non-malleable extractors, left the problem of constructing such extractors to future research, yet showed that such extractors, with great parameters, do exist. More precisely [12] proved the existence of a non-malleable extractor with seed length $d = \log(n) + O(1)$ that supports any entropy $k = \Omega(\log \log n)$, having $m = k/2 - O(1)$ output bits.

Since then, several explicit constructions of non-malleable extractors appeared in the literature, as summarized in Table 1. Moreover, different objects related to non-malleable extractors were considered in the literature as well [17, 18, 7, 1]. Up until the recent work of Chattopadhyay, Goyal, and Li [4], all constructions of non-malleable extractors worked

for entropy roughly $n/2$. The non-malleable extractor of [4] substantially improved upon previous results by supporting min-entropy $O(\log^2 n)$.

Unfortunately, unlike most previous constructions, the seed length required by the non-malleable extractor of [4] is $O(\log^2 n)$ as apposed to the desired $O(\log n)$. Thus, the exciting result of [4] sets the next natural goal at obtaining non-malleable extractors with logarithmic seed length for poly-logarithmic or even lower entropy. Besides being a natural goal, reducing the seed length to logarithmic in n is desired as in many constructions of pseudorandom objects that appear in the literature (e.g., [3, 27, 16, 21, 20, 7, 8, 5, 23, 22]) one cycles over all possible seeds of a strong seeded extractor to obtain and further process all 2^d possible outputs. Such techniques are inefficient whenever the seed length d is super-logarithmic.

2 Our Contribution

In this paper we give two constructions of non-malleable extractors that improve upon existing knowledge (see Theorem 2.1 and Theorem 2.2). Moreover, we devise several tools that we consider to be of independent interest. The first tool is an algorithm that reduces the entropy requirement of a given non-malleable extractor at the expense of slightly increasing its seed length (see Lemma 2.3). Our second algorithm increases the output length of a given non-malleable extractor from constant to optimal up to constant factors, where the constants depend only on the error guarantee (see Lemma 2.4). A third algorithm, already mentioned above, transforms a non-malleable extractor to a t-non-malleable extractor, for any desired $t > 1$ in a black-box manner (see Lemma 2.5). We now elaborate.

2.1 Two new constructions of non-malleable extractors

The first contribution of this work is a construction of a non-malleable extractor with quasi-logarithmic seed length. Our extractor also has the advantage of supporting logarithmic entropy, which is lower than that supported by the extractor of [4]. More precisely, we prove the following.

▶ **Theorem 2.1.** *There exists an explicit non-malleable extractor* $\mathsf{NMExt}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *with seed length* $d = O(\log n \cdot \log\log n)$ *for entropy* $k = \Omega(\log n)$*, having* $m = \Omega(k)$ *output bits.*

We note that Theorem 2.1 improves upon [4] both in seed length and in the required entropy. In particular, the seed length is optimal up to a multiplicative factor of $O(\log\log n)$. Our second contribution is a construction of non-malleable extractors with optimal seed length, up to a constant factor, that work for sources with entropy $n/\mathrm{polylog}\, n$. Prior to this construction, the lowest entropy supported by a non-malleable extractor with a logarithmic seed length was $0.49n$ [19]. Furthermore, even non-malleable condensers with logarithmic seed length [17] did not support sub-linear entropy.

▶ **Theorem 2.2.** *For any constant* $c > 0$ *there exists an explicit non-malleable extractor* $\mathsf{NMExt}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *with seed length* $d = O(\log n)$ *for entropy* $k = \Omega(n/\log^c n)$*, having* $\Omega(k)$ *output bits.*

In fact, the parameter c in Theorem 2.2 can be taken to be slightly super-constant so that the resulted non-malleable extractor can support entropy $k = n/(\log n)^{\omega(1)}$. This, however, will increase the seed length as it has exponential dependence in c.

2.2 Reducing the entropy requirement of a non-malleable extractor

A tool that we develop for proving Theorem 2.1 and Theorem 2.2, which we find to be of independent interest, is an algorithm that reduces the entropy requirement of a non-malleable extractor at the expense of slightly increasing its seed length. We state here a special case that is used in order to prove Theorem 2.2.

▶ **Lemma 2.3.** *There exist constants $0 < \alpha < 1 < c$ and an efficient algorithm that given a non-malleable extractor with seed length d for entropy $k = \Omega(d^{1+\alpha})$ having c output bits, produces a non-malleable extractor with seed length $O(d)$ for a lower entropy $k' = k/d^{\alpha}$.*

For a more general and formal statement, see Lemma 6.1. We are not aware of such an "entropy-seed tradeoff" being considered in previous works on seeded extractors. What is known is how to increase the output length at the expense of a longer seed. Next we consider this transformation in the context of non-malleable extractors.

2.3 Increasing the output length of a non-malleable extractor

A second tool we develop is a general method for increasing the output length of non-malleable extractors. In fact, the algorithm in the following lemma is able to increase the output length from a constant (more precisely, from $\Omega(\log(1/\varepsilon))$, where ε is the desired error guarantee) to linear in the entropy.

▶ **Lemma 2.4.** *There exists a constant c and an efficient algorithm that given a non-malleable extractor with seed length d for entropy $k = \Omega(\log n)$ and c output bits, produces a non-malleable extractor with seed length $O(d)$ for the same entropy k having $\Omega(k)$ output bits.*

A more formal statement and its proof are given in Section 7. Increasing the output length of seeded extractors is a useful tool introduced already by Nisan and Zuckerman [26]. Using the framework set in [26], Li [17] showed how to increase the output length of non-malleable extractors. However, the latter only works for high entropy sources and requires the output length one starts with to depend on the input length n. Our technique does not follow the method of Nisan and Zuckerman and involves new ideas which allows us to obtain our result.

2.4 From non-malleable extractors to t-non-malleable extractors

As mentioned, for some applications one requires an even stronger notion of non-malleability, where the output of the non-malleable extractor obtained using a typical seed is uniform even conditioned on the outputs obtained using any other t seeds for some desired parameter $t \geq 1$.

Several known constructions of non-malleable extractors are in fact t-non-malleable. Usually proving that a non-malleable extractor is a t-non-malleable extractor for some $t > 1$ is straightforward yet requires to make some changes in the proof. In other cases (e.g., [19]) one needs to make some changes in the construction itself rather than in the analysis alone.

Our next result is a black-box reduction from t-non-malleable extractors to standard (namely, $t = 1$) non-malleable extractors. Having such a reduction allows one to focus only on constructing non-malleable extractors.

▶ **Lemma 2.5.** *There exists a constant c and an efficient algorithm that given an integer $t \geq 1$ and a non-malleable extractor for entropy k with seed length d and c output bits, such that $k = \Omega(\log n + t \cdot \log(td))$, produces a t-non-malleable extractor for entropy k with seed length $O(t^2 d)$.*

A more general and formal statement and its proof appear in Section 10.

3 Proof Overview

In this section we give an informal proof overview for our results. Our techniques build on novel ideas from [4] which in turn make use of the flip-flop primitive that was introduced in [7]. To get a broad perspective, we believe it is instructive to start by describing this primitive.

3.1 The flip-flop primitive

Informally speaking, the flip-flop primitive uses a weak-source of randomness to break correlations between random variables. To this end, the flip-flop also requires an "advice bit". More precisely, a flip-flop is a function

$$\mathsf{FF}\colon \{0,1\}^n \times \{0,1\}^\ell \times \{0,1\} \to \{0,1\}^m$$

with the following property. Assume that Y, Y' are two arbitrarily correlated random variables on ℓ bit strings such that Y is uniform, and let X be an (n,k)-source that is independent of the joint distribution (Y, Y'). Then, the guarantee of the flip-flop primitive is that $\mathsf{FF}(X, Y, 0)$ looks uniform even conditioned on $\mathsf{FF}(X, Y', 1)$. Similarly, $\mathsf{FF}(X, Y, 1)$ looks uniform even conditioned on $\mathsf{FF}(X, Y', 0)$. So, informally speaking, as long as the advice bit, that is passed as the third argument to the flip-flop primitive, is different in the two applications, the flip-flop can use the weak-source X to break the correlation Y' has with Y. As mentioned, we think of the third input bit as an advice.

The construction of FF, which is implicit in [7], is based on alternating extraction – a technique that was introduced by Dziembowski and Pietrzak [14] and has found several applications in the literature since then [12, 20, 24]. We will treat FF as an atomic operation and will not get into the details of its construction here. We remark that the construction and its analysis are not very complicated. Nevertheless, we believe that thinking of FF as an atomic operation is the right level of abstraction for this discussion.

Quantitatively speaking, in [7], an explicit construction of FF was given for any n as long as $\ell = \Omega(\log n)$ and $k = \Omega(\log \ell)$, with $m = \Omega(\ell)$ output bits. In particular, if one is willing to output $O(\log n)$ bits (which usually suffices for the purpose of compositions with other pseudo-random objects), the required entropy from X is surprisingly low, namely, one only needs $k = \Omega(\log \log n)$.

3.2 Correlation breakers with advice

Informally speaking, the flip-flop primitive breaks the correlation between random variables as above, using a weak-source of randomness and an advice bit. At this point, it is not at all clear where do we expect this advice to come from when designing a non-malleable extractor. In fact, following [4], in the construction of our non-malleable extractors we will not be able to generate an advice *bit* but rather an advice *string*. More formally, we say that a function

$$\mathsf{AdvCB}\colon \{0,1\}^n \times \{0,1\}^\ell \times \{0,1\}^a \to \{0,1\}^m$$

is called a *correlation breaker with advice* if for any two ℓ-bit random variables Y, Y' such that Y is uniform and for any independent (n,k)-source X, it holds that $\mathsf{FF}(X, Y, \alpha)$ looks uniform even conditioned on $\mathsf{FF}(X, Y', \alpha')$ for any distinct $\alpha, \alpha' \in \{0,1\}^a$ (for a formal definition the reader is referred to Definition 4.11).

Note that a correlation breaker with advice of length $a = 1$ is exactly the flip-flop primitive. Clearly, it is easier to generate long advices than shorter ones. Nevertheless, one

can implement an AdvCB using the flip-flop primitive. We will not delve into the details of the reduction here, and will be satisfied by stating that this reduction, as was done implicity in [7, 4], works for every n, a with $\ell = \Omega(a \cdot \log(an))$, $k = \Omega(a \cdot \log(a \log n))$, and has $m = \Omega(\log n)$ output bits (see Theorem 4.12 for a formal statement).

In fact, as in [4], we will need a somewhat stronger guarantee. Namely, not only $\mathsf{AdvCB}(X, Y, \alpha)$ should be uniform even conditioned on $\mathsf{AdvCB}(X, Y', \alpha')$ with the notation set as above, but rather $\mathsf{AdvCB}(X, Y, \alpha)$ should look uniform even after given $\mathsf{AdvCB}(X', Y', \alpha')$, where X' may correlate arbitrarily with the (n, k)-source X, as long as the joint distribution (X, X') is independent of the joint distribution (Y, Y').

3.3 The [4] reduction from non-malleable extractors to advice generators

In this section we introduce the notion of an *advice generator* that is implicit in [4], and present the novel reduction by [4] from non-malleable extractors to advice generators. In the following section we introduce our improved reduction. We start by defining the notion of an *advice generator* (for a formal treatment, see Definition 5.1). A function

$$\mathsf{AdvGen}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^a$$

is called an advice generator if for any X, Y as above and for any function $\mathcal{A}\colon \{0,1\}^d \to \{0,1\}^d$ with no fixed points, it holds that $\mathsf{AdvGen}(x, y) \neq \mathsf{AdvGen}(x, \mathcal{A}(y))$ with high probability over $x \sim X$, $y \sim Y$. The general idea in [4] is to compute an advice using x, y and feed that advice to a correlation breaker with advice. Namely, given an advice generator AdvGen and a correlation breaker with advice AdvCB, the non-malleable extractor is defined as

$$\mathsf{NMExt}(x, y) = \mathsf{AdvCB}(x, y, \mathsf{AdvGen}(x, y)). \tag{1}$$

Indeed, with high probability, the advices $\mathsf{AdvGen}(X, Y)$ and $\mathsf{AdvGen}(X, \mathcal{A}(Y))$ are distinct, and so one may carelessly conclude that AdvCB guarantees that $\mathsf{NMExt}(X, Y)$ is uniform even conditioned on $\mathsf{NMExt}(X, \mathcal{A}(Y))$. Of course, the problem with this argument is that there are correlations between the advices and between X, Y.

To salvage the argument above, one needs to make sure that even conditioned on the fixings of the advices $\mathsf{AdvGen}(X, Y)$, $\mathsf{AdvGen}(X, \mathcal{A}(Y))$, it holds that X and Y remain independent. So there is a strong limitation on the type of computation that can be carried by AdvGen. Even having such a guarantee there are a couple of problems with such a general method for constructing a non-malleable extractor. First, we must make sure that conditioned on the fixings of $\mathsf{AdvGen}(X, Y)$, $\mathsf{AdvGen}(X, \mathcal{A}(Y))$, it holds that X has enough entropy as required by AdvCB. Typically, this is a non-issue. Second, we need Y to remain uniform even after these fixings. Nevertheless, by constructing an advice generator that has a suitable interplay with AdvCB, a construction having the general form above was used by [4] for their construction of non-malleable extractors.

Quantitatively speaking, [4] constructed an advice generator with advice length $a = O(\log n)$ (see Section 8.1) that, using the reduction above, can be shown to yield a non-malleable extractor for min-entropy $\Omega(\log^2 n)$ with seed length $O(\log^2 n)$. In the next section we describe our improved reduction from non-malleable extractors to advice generators.

3.4 An improved reduction

We now present a different way of constructing a non-malleable extractor given an advice generator. Our reduction will enable us to obtain non-malleable extractors with shorter seeds that work for lower min-entropies compared to [4].

A building block that we use is the seeded extractor of Raz [28] that works with weak-seeds. This is a strong seeded extractor $\mathsf{Raz}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ that has the same guarantee as standard strong seeded extractors even if the seed is not uniform, but rather has min-entropy $0.6d$. Raz [28] gave an explicit construction of such an extractor with seed length $d = O(\log n)$ that supports any entropy $k = \Omega(d)$. See Theorem 4.3 for a formal statement.

With this building block, we are ready to define our reduction. First, we divide the seed y to 3 parts $y = y_1 \circ y_2 \circ y_3$, where y_i has length d_i. We only assume that d_1 is very small compared to d (taking $d_1 \leq d/1000$ will do) and set $9d_2 = d_3$. Our reduction make use of an advice generator $\mathsf{AdvGen}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^a$ that has the following extra guarantee. For any function $\mathcal{A}\colon \{0,1\}^d \to \{0,1\}^d$ with no fixed points, it holds that with high probability over the fixing of $\mathsf{AdvGen}(X,Y)$, $\mathsf{AdvGen}(X,\mathcal{A}(Y))$:

- The random variables X, Y remain independent;
- X has not lost more than, say, half of its min-entropy;
- The random variable $Y_2 \circ Y_3$ has min-entropy rate 0.99.

Given any such "nice" advice generator, we define our non-malleable extractor by

$$\mathsf{NMExt}(x,y) = \mathsf{AdvCB}\left(y_3, \mathsf{Raz}(x,y_2), \mathsf{AdvGen}(x,y)\right). \tag{2}$$

It is worthwhile to compare the above definition with the reduction given by Equation (1). The most important difference being the "switch" that was done between the roles of the source and the seed. Namely, the seed Y to the non-malleable extractor takes the role of a source in AdvCB as (a suffix of) it is being passed as the first argument, whereas the seed to AdvCB is this function $\mathsf{Raz}(X,Y_2)$ of both X and Y. This switch is what makes the reduction more efficient in the sense that the resulted non-malleable extractor has a shorter seed and can support a lower entropy. Informally speaking, the reason for this is that Y_3 makes a much shorter source than X as the latter consists of n bits whereas we will end up setting Y to have length which is logarithmic in n.

3.4.1 Analyzing the reduction

We now give a sketch of the analysis for the reduction given by Equation (2). First, according to the definition of AdvGen, by aggregating a small error, we may assume that $\alpha = \mathsf{AdvGen}(X,Y)$ and $\alpha' = \mathsf{AdvGen}(X,\mathcal{A}(Y))$ are distinct fixed strings. Further, the three extra properties of AdvGen hold.

By the third property, $Y_2 \circ Y_3$ has min-entropy rate 0.99. Since $d_2 = (d_2 + d_3)/10$, we argue that with high probability over Y_3, it holds that Y_2 has min-entropy rate 0.9. To see why a claim of this sort should be true, think of the special case where 0.99 fraction of the bits of $Y_2 \circ Y_3$ are distributed uniformly and independently at random, and the remaining 0.01 fraction of the bits behave adversarially. Since Y_2 is a block of density 0.1 in $Y_2 \circ Y_3$, even in the worst case where Y_2 contains all the "bad" bits, their fraction within Y_2 is at most $0.01/0.1 = 0.1$, and so 0.9 fraction of the bits in Y_2 are uniform and independent of each other, leaving Y_2 with min-entropy rate of 0.9. A somewhat more careful argument can be carried out to handle the more general case where we only assume that the min-entropy rate of $Y_2 \circ Y_3$ is 0.99.

Once we have established that Y_2 has min-entropy rate 0.9, we have that $\mathsf{Raz}(X,Y_2)$ is close to uniform. For this we use the guarantees that the entropy of X remained high after the fixings of the advices, and that these fixings have not introduced correlations between X and Y_2. In fact, since Raz is strong, with high probability over the fixing of $y_2 \sim Y_2$ we have

that $\mathsf{Raz}(X, y_2)$ is close to uniform. Since $\mathsf{Raz}(X, y_2)$ is a deterministic function of X, we can further fix $\mathcal{A}(Y)_2$ without affecting $\mathsf{Raz}(X, y_2)$ and without introducing correlations between X, Y. One can then show that these fixings of Y_2 and $\mathcal{A}(Y)_2$ can only reduce the min-entropy of Y_3 by roughly $2d_2$, and so the min-entropy of Y_3 is at least $0.99(d_2 + d_3) - 2d_2 > 0.8d_3$. Namely, Y_3 is a $(d_3, 0.8d_3)$-source.

Note that by now, $\mathsf{Raz}(X, Y_2)$ and $\mathsf{Raz}(X, \mathcal{A}(Y)_2)$ are deterministic functions of X whereas Y_3, $\mathcal{A}(Y)_3$ are independent of X. Further, $\mathsf{Raz}(X, y_2)$ is close to uniform and Y_3 is has min-entropy rate 0.8. Thus, the hypothesis of AdvCB is met and we conclude that $\mathsf{NMExt}(X, Y)$ looks uniform even conditioned on $\mathsf{NMExt}(X, \mathcal{A}(Y))$, as desired.

3.5 Reducing the entropy requirement of non-malleable extractors

In this section we describe another contribution of this paper stated as Lemma 2.3, which is a black-box transformation that given a non-malleable extractor with seed length d for entropy $k = \Omega(d^{1+\alpha})$, produces a non-malleable extractor for a lower entropy $k' = k/d^{\alpha}$ with seed length $O(d)$. Here $\alpha > 0$ is some small universal constant. Our reduction is composed of two steps. In the first step we construct an advice generator for entropy k' given a non-malleable extractor for entropy k. We then apply our reduction from Section 3.4 to obtain a non-malleable extractor for entropy k' using the advice generator.

To describe this "reversed" reduction, namely, the reduction from advice generators to non-malleable extractors with higher entropy, we make use of several building blocks, the first of which is a somewhere condenser. Informally speaking, this is a sequence of functions $\{\mathsf{Cond}_i \colon \{0,1\}^n \to \{0,1\}^n\}_{i=1}^r$ with the following property. Let $\delta > 0$. Then, for any $(n, \delta k)$-source X there exists $g \in [r]$ such that $\mathsf{Cond}_g(X)$ is an (n, k)-source. It is known [2, 33] how to construct such a somewhere condenser with $r = \mathrm{poly}(1/\delta)$ (see Theorem 4.5). Building on [4], we also make use of a strong seeded extractor Ext and a binary error correcting code ECC with relative distance, say, $1/4$ having a constant rate.

Given these building blocks, say we are given a non-malleable extractor $\mathsf{NMExt} \colon \{0,1\}^n \times \{0,1\}^{d_1} \to \{0,1\}^{\log m}$ for entropy k, where m will be set later on. Our advice generator is defined as follows. Split the seed y to two substrings $y = y_1 \circ y_2$, where y_1 is of length d_1 and $d_2 = 100d_1$. We define

$$\mathsf{AdvGen}(x, y) = \mathsf{NMExt}(\mathsf{Cond}_1(x), y_1) \circ \cdots \circ \mathsf{NMExt}(\mathsf{Cond}_r(x), y_1) \circ \mathsf{ECC}(y_2)_{\mathsf{Ext}(x, y_1)},$$

where we interpret the output of $\mathsf{Ext}(x, y_1)$ as a size $\log m$ subset of the index set $[D_2]$ and use $\mathsf{ECC}(y_2)_{\mathsf{Ext}(x,y_1)}$ to denote the projection of the string $\mathsf{ECC}(y_2)$ on to that set of indices. Note that for this we need the output of Ext to consists of $O(\log m \cdot \log d_1)$ bits.

The construction above is influenced by the advice generator construction of [4]. In particular, with the notation set above, the advice generator of [4] can be written as $\mathsf{AdvGen}(x, y) = y_1 \circ \mathsf{ECC}(y_2)_{\mathsf{Ext}(x,y_1)}$ (see Section 8.1).

3.5.1 Analyzing the entropy reduction transformation

In this section we give an informal analysis showing that the function AdvGen is indeed an advice generator for entropy δk. To this end we consider an $(n, \delta k)$-source X and a function $\mathcal{A} \colon \{0,1\}^d \to \{0,1\}^d$ with no fixed points. We start by fixing $y_1 \sim Y_1$ and $y_1' \sim \mathcal{A}(Y)_1$ and consider two cases according to whether or not $y_1 = y_1'$.

Case 1 – $y_1 = y_1'$. In this case, following [4], we show that with high probability

$$\mathsf{ECC}(Y_2)_{\mathsf{Ext}(X,y_1)} \neq \mathsf{ECC}(\mathcal{A}(Y)_2)_{\mathsf{Ext}(x,y_1)},$$

which in particular will guarantee that with high probability $\mathsf{AdvGen}(X, Y) \neq \mathsf{AdvGen}(X, \mathcal{A}(Y))$ in this case. To see why this is true, note that since $y_1 = y_1'$ we have that $Y_2 \neq \mathcal{A}(Y)_2$ and so the two codewords $\mathsf{ECC}(Y_2)$, $\mathsf{ECC}(\mathcal{A}(Y)_2)$ agree on at most $3/4$ of the indices. In particular, by projecting each of these codewords to a random set of indices of size $\log m$, we will get the same string with a probability bound that decrease polynomially with $1/m$. Of course, we do not (and cannot) sample a truly uniform projection. Nevertheless, as Ext is a strong seeded extractor, for most fixings of y_1 it holds that $\mathsf{Ext}(X, y_1)$ is close to a random subset of $[D_2]$ which suffices for the argument above to go through.

Case 2 – $y_1 \neq y_1'$. For analyzing this case, we recall that NMExt is a non-malleable extractor for entropy k and that there is some $g \in [r]$ for which $\mathsf{Cond}_g(X)$ has min-entropy k. Using the dichotomic point of view on non-malleable extractors (see Lemma 4.14), one can show that $\mathsf{NMExt}(\mathsf{Cond}_g(X), y_1)$ is close to uniform even conditioned on $\mathsf{NMExt}(\mathsf{Cond}_g(X), y_1')$ for most choices of y_1. In particular, the probability that the two strings $\mathsf{NMExt}(\mathsf{Cond}_g(X), y_1)$, $\mathsf{NMExt}(\mathsf{Cond}_g(X), y_1')$ are equal is polynomially small in m.

By taking $m = \mathrm{poly}(1/\varepsilon)$ we can bound the error of AdvGen by ε. This choice of m yields an advice length $a = O(r \cdot \log(1/\varepsilon)) = \mathrm{poly}(1/\delta) \cdot \log(1/\varepsilon)$.

So far we gave an informal proof showing that AdvGen is an advice generator for entropy δk. Recall that to use the advice generator in our reduction from Section 3.4, AdvGen must have some extra guarantees. Perhaps the most subtle of which is that conditioned on the fixings of $\mathsf{AdvGen}(X, Y)$, $\mathsf{AdvGen}(X, \mathcal{A}(Y))$, the random variables X, Y must remain independent. We assure the reader that this is the case with our construction due to the "alternating" fashion of the computation involving AdvGen, though we will skip the details in this proof overview and refer the reader to Section 6.

3.6 Increasing the output length of a non-malleable extractor

In this section we briefly describe our algorithm that increases the output length of a given non-malleable extractor described in Lemma 2.4. Being a bit more formal, for a desired error guarantee ε, we show how to increase the output length of a non-malleable extractor NMExt from $O(\log(1/\varepsilon))$ to $\Omega(k/\log(1/\varepsilon))$. Here, k is the entropy supported by NMExt. As with our entropy reduction transformation described in Section 3.5, here too the general idea is to use the given non-malleable extractor NMExt so to obtain an advice generator AdvGen which in turn will be used to construct the desired non-malleable extractor NMExt' using our reduction from Section 3.4. More precisely, borrowing notation from previous sections, we define

$$\mathsf{AdvGen}(x, y) = \mathsf{NMExt}(x, y_1) \circ \mathsf{ECC}(y_2)_{\mathsf{Ext}(x, y_1)}.$$

A similar argument to the one used in Section 3.5 shows that AdvGen is an advice generator for entropy k (in fact, this is a special case of the argument from Section 3.5). In particular, we show that if one aims for an error guarantee ε, it suffices that the output length of NMExt consists of $O(\log(1/\varepsilon))$ bits. At this point we can apply the reduction from Section 3.4 to AdvGen. This results in a non-malleable extractor NMExt' that supports the same entropy k, though it has the advantage of having output length $\Omega(k/\log(1/\varepsilon))$.

3.7 Proof overview for Theorem 2.1 and Theorem 2.2

In this section we give an overview for the proofs of Theorem 2.1 and Theorem 2.2, starting with the first theorem. As our starting point, we apply our improved reduction given in

Section 3.4 with the advice generator of [4]. This already yields a non-malleable extractor with seed length $O(\log n \cdot \log\log n)$ that supports entropy $\Omega(\log n \cdot \log\log n)$. Our second step is to apply the entropy reduction transformation so to obtain a second non-malleable extractor that supports a lower entropy. By choosing the parameters correctly, one can show that the resulted non-malleable extractor can support entropy $\Omega(\log n)$ while maintaining a seed of length $O(\log n \cdot \log\log n)$. As our final step we apply the transformation for increasing the output length that was described in the previous section to yield Theorem 2.1.

For the proof of Theorem 2.2, our starting point is any of the constructions of non-malleable extractors for entropy $0.6n$ with seed length $O(\log n)$ (e.g., the one given in Theorem 4.6) and denote this non-malleable extractor by NMExt_0. We now apply the entropy reduction transformation described in Section 3.5 to NMExt_0 so to obtain a new non-malleable extractor, which we denote by NMExt_1. Working out the parameters, NMExt_1 can be shown to support entropy $n/(\log n)^{\alpha}$ for some small universal constant $\alpha > 0$. Further, NMExt_1 has seed length $d_1 = O(d) = O(\log n)$.

We continue by applying the entropy reduction transformation again, this time to NMExt_1 and obtain a new non-malleable extractor NMExt_2 that has seed length $O(d_1) = O(\log n)$ and supports entropy $n/(\log n)^{2\alpha}$. By repeating this process, we construct a sequence of non-malleable extractors where each extractor supports lower entropy than its predecessor. After r steps, we obtain a non-malleable extractor NMExt_r that supports entropy $n/(\log n)^{\alpha r}$ and has seed length $2^{O(r)} \cdot \log n$. The proof of Theorem 2.2 follows by taking $r = c/\alpha$, where c is the desired constant.

3.8 From non-malleable extractor to t-non-malleable extractors

We turn to say a few words about our reduction from non-malleable extractors to t-non-malleable extractors for any $t > 1$ stated in Lemma 2.5. Recall that our construction of non-malleable extractors from Section 3.4 consists of two steps. First, we construct a "nice" advice generator. Second, the generated advice is passed to a correlation breaker with advice.

One can generalize the notions of advice generators and correlation breakers with advice to t-advice generators and t-correlation breakers with advice in the natural way for any $t \geq 1$. One can then show that the idea presented in Section 3.4 of constructing non-malleable extractors based on advice generators and correlation breakers with advice can be extended to any $t > 1$. Namely, given a t-advice generator and t-correlation breaker with advice, one can obtain a t-non-malleable extractor using the exact same reduction (see Lemma 10.4).

We already know how to construct a t-correlation breaker with advice (see Theorem 4.12) for any $t \geq 1$. A key observation we make is that for any $t \geq 1$ one can construct a t-advice generator using a standard non-malleable extractor. This allows us to reduce the problem of constructing t-non-malleable extractors to the problem of constructing non-malleable extractors. For further details see Section 10.

4 Preliminaries

Unless stated otherwise, the logarithm in this paper is always taken base 2. For every natural number $n \geq 1$, define $[n] = \{1, 2, \ldots, n\}$. Throughout the paper we avoid the use of floor and ceiling in order not to make the equations cumbersome.

Random variables and distributions

We sometimes abuse notation and syntactically treat random variables and their distribution as equal, specifically, we denote by U_m a random variable that is uniformly distributed over $\{0,1\}^m$. Furthermore, if U_m appears in a joint distribution (U_m, X) then U_m is independent of X. When m is clear from context, we omit it from the subscript and write U.

Let X, Y be two random variables. We say that Y is a *deterministic function of* X if the value of X determines the value of Y. Namely, there exists a function f such that $Y = f(X)$. Let $X, Y, Z_1, \ldots, Z_r$ be random variables. We use the following shorthand notation and write $(X, Z_1, \ldots, Z_r) \approx_\varepsilon (Y, \cdot)$ for $(X, Z_1, \ldots, Z_r) \approx_\varepsilon (Y, Z_1, \ldots, Z_r)$.

Statistical distance

The *statistical distance* between two distributions X, Y on the same domain D is defined by $\mathsf{SD}(X,Y) = \max_{A \subseteq D} \{|\Pr[X \in A] - \Pr[Y \in A]|\}$. If $\mathsf{SD}(X,Y) \leq \varepsilon$ we write $X \approx_\varepsilon Y$ and say that X and Y are ε-close.

Min-entropy

The *min-entropy* of a random variable X, denoted by $H_\infty(X)$, is defined by $H_\infty(X) = \min_{x \in \mathsf{supp}(X)} \log_2(1/\Pr[X = x])$. If X is supported on $\{0,1\}^n$, we define the *min-entropy rate* of X by $H_\infty(X)/n$. In such case, if X has min-entropy k or more, we say that X is an (n,k)-source.

Pseudorandom objects we use

▶ **Definition 4.1** (Seeded extractors)**.** A function $\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is called a *seeded extractor* for entropy k, with error ε, if for any (n,k)-source X it holds that $\mathsf{Ext}(X,S) \approx_\varepsilon U_m$, where S is uniformly distributed over $\{0,1\}^d$ and is independent of X. We say that Ext is a *strong seeded-extractor* if $(\mathsf{Ext}(X,S), S) \approx_\varepsilon (U_m, U_d)$.

Throughout the paper we make use of the following explicit pseudorandom objects.

▶ **Theorem 4.2** ([15])**.** *There exists a universal constant $c > 0$ such that the following holds. For all positive integers n, k and $\varepsilon > 0$, there exists an efficiently-computable strong seeded-extractor $\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ for entropy k, with error ε, seed length $d = c \cdot \log(n/\varepsilon)$, and $m = 0.99 \cdot k$ output bits.*

▶ **Theorem 4.3** ([28])**.** *For all integers n, k, d and for any $\varepsilon > 0$ such that $d = \Omega(\log(n/\varepsilon))$ and $k = \Omega(d)$, there exists an efficiently-computable function $\mathsf{Raz}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{k/2}$ with the following property. Let X be an (n,k)-source, and let Y be an independent $(d, 0.6d)$-source. Then, $(\mathsf{Raz}(X,Y), Y) \approx_\varepsilon (U, Y)$.*

▶ **Theorem 4.4.** *There exists a universal constant c such that the following holds. For all integers n, there exists an explicit error correcting code $\mathsf{ECC}\colon \{0,1\}^n \to \{0,1\}^{cn}$ with relative distance $1/4$.*

▶ **Theorem 4.5** ([2, 33])**.** *For any integer n and any $\delta > 0$ there exists a sequence of efficiently computable functions $\{\mathsf{Cond}_i\colon \{0,1\}^n \to \{0,1\}^m\}_{i=1}^{\Delta}$ with $\Delta = \mathrm{poly}(1/\delta)$ and $m = n \cdot \mathrm{poly}(\delta)$ such that the following holds. For any $(n, \delta n)$-source X, the joint distribution of $\{\mathsf{Cond}_i(X)\}_{i=1}^r$ is $2^{-\Omega(\delta^2 n)}$-close to a convex combination such that for any participant $(Y_1, \ldots, Y_r)$ in the combination, there exists $g \in [\Delta]$ such that Y_g has min-entropy rate 0.6.*

▶ **Theorem 4.6** ([9]). *For all integers n, m, t such that $n = \Omega(mt)$ and for any $\varepsilon > 0$ there exists a $\mathrm{poly}(n)$-time computable t-non-malleable extractor $\mathsf{CRS}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ for $(n, 0.6n)$-sources, with error 2^{-m} and seed length $d = tm + 2\log n$.*

Average conditional min-entropy

We make use of the notion of average min-entropy and some basic properties of it.

▶ **Definition 4.7.** Let X, W be two random variables. The *average conditional min-entropy* of X given W is defined as $\widetilde{H}_\infty(X \mid W) = -\log_2\left(\mathbf{E}_{w\sim W}\left[\max_x \Pr\left[X = x \mid W = w\right]\right]\right)$.

▶ **Lemma 4.8** ([11]). *Let X, Y, Z be random variables such that Y has support size at most 2^ℓ. Then, $\widetilde{H}_\infty(X \mid (Y, Z)) \geq \widetilde{H}_\infty(X \mid Z) - \ell$. In particular, $\widetilde{H}_\infty(X \mid Y) \geq H_\infty(X) - \ell$.*

▶ **Lemma 4.9** ([11]). *For any two random variables X, Y and any $\varepsilon > 0$, it holds that*

$$\Pr_{y\sim Y}\left[H_\infty(X \mid Y = y) < \widetilde{H}_\infty(X \mid Y) - \log(1/\varepsilon)\right] \leq \varepsilon.$$

We further make use of the following simple lemma.

▶ **Lemma 4.10.** *Let X, Y, Z be random variables such that for any $y \in \mathsf{supp}(Y)$, the random variables $(X \mid Y = y)$ and $(Z \mid Y = y)$ are independent. Assume that X is supported on $\{0,1\}^n$. Then, $\mathsf{SD}\left((X, Y, Z), (U_n, Y, Z)\right) = \mathsf{SD}\left((X, Y), (U_n, Y)\right)$.*

4.1 Correlation breakers with advice

In this section we introduce the notion of *correlation breakers with advice* which is a variant of local correlation breakers [7] that is implicit in [4]. We then state the parameters of the explicit construction obtained by following the proof of [4].

▶ **Definition 4.11.** For an integer $t \geq 1$ a *t-correlation-breaker with advice* for entropy k and error ε is a function

$$\mathsf{AdvCB}\colon \{0,1\}^w \times \{0,1\}^\ell \times \{0,1\}^a \to \{0,1\}^m$$

with the following property. Let $X^0, X^1, \ldots, X^t$ be random variables distributed over $\{0,1\}^w$ such that X^0 has min-entropy k. Let $Y^0, Y^1, \ldots, Y^t$ be random variables over $\{0,1\}^\ell$ that are jointly independent of $(X^0, X^1, \ldots, X^t)$ such that Y^0 is uniform. Then, for any strings $s^0, s^1, \ldots, s^t \in \{0,1\}^a$ such that $s^0 \notin \{s^1, \ldots, s^t\}$, it holds that

$$\left(\mathsf{AdvCB}(X^0, Y^0, s^0), \{\mathsf{AdvCB}(X^i, Y^i, s^i)\}_{i=1}^t\right) \approx_\varepsilon (U_m, \cdot)\,.$$

The third argument to the function AdvCB is called the *advice*.

▶ **Theorem 4.12.** *For all integers ℓ, w, a, t and for any $\varepsilon \in (0,1)$ such that*

$$\ell = \Omega\left(at \cdot \log\left(\frac{aw}{\varepsilon}\right)\right), \tag{3}$$

there exists a $\mathrm{poly}(\ell, w)$-time computable t-correlation-breaker with advice $\mathsf{AdvCB}\colon \{0,1\}^w \times \{0,1\}^\ell \times \{0,1\}^a \to \{0,1\}^m$, for entropy

$$k = \Omega\left(at \cdot \log\left(\frac{a\ell}{\varepsilon}\right)\right), \tag{4}$$

with error ε and $m = \Omega(\ell/(at))$ output bits.

4.2 An equivalent definition for t-non-malleable extractors

In this section we give an equivalent definition for t-non-malleable extractors. We make use of this equivalence in some of our proofs, and in general, we find this alternative definition to be more convenient to work with than the original definition of non-malleable extractors.

▶ **Definition 4.13** (Dichotomic t-non-malleable extractors)**.** A function $\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is called a *dichotomic t-non-malleable extractor* for entropy k with error ε if for any (n,k)-source X there exists a set $B \subset \{0,1\}^d$ of size at most $\varepsilon \cdot 2^d$ such that the following holds. For any $y \notin B$ and any $y^1, \ldots, y^t \in \{0,1\}^d \setminus \{y\}$ it holds that

$$\left(\mathsf{Ext}(X,y), \left\{\mathsf{Ext}(X,y^i)\right\}_{i=1}^t\right) \approx_\varepsilon (U_m, \cdot)\,.$$

The following simple lemma that shows the equivalence between the definition of non-malleable extractors and dichotomic non-malleable extractors, up to some loss in the error guarantee, builds on ideas by [5].

▶ **Lemma 4.14.** *Let* $\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *be some function.*

- *If* Ext *is a t-non-malleable extractor for entropy k with error ε then* Ext *is a dichotomic t-non-malleable extractor for entropy k with error $\sqrt{\varepsilon}$.*
- *If* Ext *is a dichotomic t-non-malleable extractor for entropy k with error ε then* Ext *is a t-non-malleable extractor for entropy k with error 2ε.*

Proof. We start by proving the first item. Let X be an (n,k)-source. Define B to be the set of all $y \in \{0,1\}^d$ for which there exist $y^1, \ldots, y^t \in \{0,1\}^d \setminus \{y\}$ such that

$$\left(\mathsf{Ext}(X,y), \{\mathsf{Ext}(X,y^i)\}_{i=1}^t\right) \not\approx_{\sqrt{\varepsilon}} (U_m, \cdot)\,. \tag{5}$$

We want to prove that $\beta \triangleq |B|/2^d \leq \sqrt{\varepsilon}$. To this end, for every $i \in [t]$ define the function $\mathcal{A}_i\colon \{0,1\}^d \to \{0,1\}^d$ as follows. For $y \notin B$ define $A_i(y)$ arbitrarily, only ensuring that there are no fixed points. For $y \in B$, let $y^1, \ldots, y^t \in \{0,1\}^d \setminus \{y\}$ be a sequence of seeds for which Equation (5) holds, and set $A_i(y) = y^i$. Note that by the definition of B,

$$\left(\mathsf{Ext}(X,Y), \{\mathsf{Ext}(X,\mathcal{A}_i(Y))\}_{i=1}^t\right) \not\approx_{\beta\cdot\sqrt{\varepsilon}} (U_m, \cdot)\,.$$

On the other hand, as Ext is a t-non-malleable extractor with error ε

$$\left(\mathsf{Ext}(X,Y), \{\mathsf{Ext}(X,\mathcal{A}_i(Y))\}_{i=1}^t\right) \approx_\varepsilon (U_m, \cdot)\,,$$

which concludes the proof of the first item.

As for the second item, let $\mathcal{A}_1, \ldots, \mathcal{A}_t\colon \{0,1\}^d \to \{0,1\}^d$ be functions with no fixed points, and let X be an (n,k)-source. As Ext is a dichotomic t-non-malleable extractor for entropy k with error ε, there exists a set $B \subset \{0,1\}^d$ of size $|B| \leq \varepsilon \cdot 2^d$ such that for any $y \notin B$ it holds that

$$\left(\mathsf{Ext}(X,y), \{\mathsf{Ext}(X,\mathcal{A}_i(y))\}_{i=1}^t\right) \approx_\varepsilon (U_m, \cdot)\,.$$

As $|B| \leq \varepsilon \cdot 2^d$ we conclude that

$$\left(\mathsf{Ext}(X,Y), \{\mathsf{Ext}(X,\mathcal{A}_i(Y))\}_{i=1}^t\right) \approx_{2\varepsilon} (U_m, \cdot)\,,$$

which concludes the proof of the second item. ◀

5 A New Reduction From Non-Malleable Extractors to Advice Generators

In this section we describe our reduction from non-malleable extractors to advice generators. Most of our results make use of this reduction by plugging in different advice generators (some of which are constructed using other non-malleable extractors). We start by giving a formal definition of advice generators.

▶ **Definition 5.1** (Advice generators)**.** A function $\mathsf{AdvGen}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^a$ is called an *advice generator* for entropy k with error ε if the following holds. For any (n,k)-source X, an independent random variable Y that is uniform over $\{0,1\}^d$, and a function $\mathcal{A}\colon \{0,1\}^d \to \{0,1\}^d$ with no fixed points, it holds that

$$\Pr_{\substack{x\sim X\\ y\sim Y}}[\mathsf{AdvGen}(x,y) = \mathsf{AdvGen}(x,\mathcal{A}(y))] \leq \varepsilon.$$

The second input to AdvGen is called the *seed*.

For our reduction to work, some extra guarantee is needed from the advice generator. Informally speaking, it is required that even conditioned on the fixings of the advices, the random variables X, Y remain independent and have a sufficient amount of entropy. The formal guarantee is encapsulated in the following definition.

▶ **Definition 5.2** (Nice advice generators)**.** An advice generator $\mathsf{AdvGen}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^a$ for entropy k with error ε is said to be d_1-nice if the following holds. Let X be an (n,k)-source, let Y be a random variable that is independent of X and is uniformly distributed over $\{0,1\}^d$, and let $\mathcal{A}\colon \{0,1\}^d \to \{0,1\}^d$ be a function with no fixed points. Then, except with probability ε over the fixings of $\mathsf{AdvGen}(X,Y)$, $\mathsf{AdvGen}(X,\mathcal{A}(Y))$ it holds that:

- X, Y are independent.
- $H_\infty(X) \geq 0.99k$.
- The length $d - d_1$ suffix of Y has min-entropy rate 0.99.

In the following lemma we present our reduction from non-malleable extractors to nice advice generators.

▶ **Lemma 5.3.** *There exist universal constants $0 < c < 1 < c', c''$ such that the following holds. Let $\mathsf{AdvGen}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^a$ be an explicit advice generator for entropy k with error ε that is d_1-nice, with $d_1 \leq d/2$. Then, for any integer m such that*

$$m \leq c \cdot k/a$$
$$d \geq c' \cdot \max\left(a \cdot \log\left(\frac{am}{\varepsilon}\right), \log(n/\varepsilon)\right),$$
$$k \geq c'' \cdot \max\left(a \cdot \log\left(\frac{ad}{\varepsilon}\right), \log(n/\varepsilon)\right),$$

there exists an explicit non-malleable extractor $\mathsf{NMExt}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ for entropy k, with error $O(\sqrt{\varepsilon})$.

Proof. We start by describing the construction of NMExt and then turn to the analysis. Given a string $y \in \{0,1\}^d$, we partition y to three consecutive substrings $y = y_1 \circ y_2 \circ y_3$, where $|y_1| = d_1$, $|y_2| = d_2 = \Omega(\log(n/\varepsilon))$ is a sufficient length for a seed of the extractor from Theorem 4.3 set with error ε, and $|y_3| = d_3 = 9d_2$. By choosing a sufficiently large constant c', d is large enough so to satisfy these properties. We make use of the following building blocks:

- Let $\mathsf{Raz}\colon \{0,1\}^n \times \{0,1\}^{d_2} \to \{0,1\}^\ell$ be the extractor from Theorem 4.3, where

  $$\ell = c''' \cdot \max\left(am, a\log(ad/\varepsilon)\right)$$

  for some suitable constant c''' to be chosen next. By our choice of d_2, the error of Raz is bounded above by ε.
- Let $\mathsf{AdvCB}\colon \{0,1\}^{d_3} \times \{0,1\}^\ell \times \{0,1\}^a \to \{0,1\}^m$ be the correlation breaker with advice from Theorem 4.12 set with error ε. By Theorem 4.12, the constant c''' can be chosen such that the output length of AdvCB is indeed m.

  With the notation set and using the building blocks above, we define

  $$\mathsf{NMExt}(x,y) = \mathsf{AdvCB}\left(y_3, \mathsf{Raz}(x,y_2), \mathsf{AdvGen}(x,y)\right).$$

We now turn to the analysis. Let X be an (n,k)-source, let Y be an independent random variable that is uniformly distributed over $\{0,1\}^d$, and let $\mathcal{A}\colon \{0,1\}^d \to \{0,1\}^d$ be a function with no fixed points. As AdvGen is a d_1-nice advice generator with error ε, we have that except with probability ε over the fixings $\alpha \sim \mathsf{AdvGen}(X,Y)$, $\alpha' \sim \mathsf{AdvGen}(X,\mathcal{A}(Y))$, it holds that

- $\alpha \neq \alpha'$.
- X, Y remain independent.
- $H_\infty(X) \geq 0.99k$.
- The length $d - d_1$ suffix of Y has min-entropy rate 0.99.

We condition on such fixing. Next, we argue that except with probability ε over $y_3 \sim Y_3$ it holds that $Y_2 \mid (Y_3 = y_3)$ has min-entropy rate at least 0.6. To see this, apply Lemma 4.8 to obtain

$$\widetilde{H}_\infty(Y_2 \circ Y_3 \mid Y_3) \geq H_\infty(Y_2 \circ Y_3) - |Y_3| \geq 0.99(d_2+d_3) - d_3 = 0.9d_2.$$

Thus, by Lemma 4.9, except with probability ε over $y_3 \sim Y_3$ it holds that

$$H_\infty(Y_2 \mid Y_3 = y_3) = H_\infty(Y_2 \circ Y_3 \mid Y_3 = y_3) \geq 0.9d_2 - \log(1/\varepsilon) \geq 0.6d_2.$$

Therefore, except with probability ε over the fixing of Y_3, the min-entropy rate of Y_2 is bounded below by 0.6. For the remaining of the proof we assume that the min-entropy rate of Y_2 is at least 0.6, and aggregate an additional error of ε to the total error.

By setting the constant c'' to be large enough, we can guarantee that $H_\infty(X) \geq 0.99k \geq 2\ell$ and that $H_\infty(X) = \Omega(d_2)$. Since Y_2 is a $(d_2, 0.6d_2)$-source with $d_2 = \Omega(\log(n/\varepsilon))$, we can apply Theorem 4.3 and conclude that

$$(\mathsf{Raz}(X,Y_2), Y_2) \approx_\varepsilon (U_\ell, Y_2).$$

As $\mathsf{Raz}(X,Y_2)$ is independent of $\mathcal{A}(Y)_2$ conditioned on the fixing of Y_2, Lemma 4.10 implies that

$$(\mathsf{Raz}(X,Y_2), Y_2, \mathcal{A}(Y)_2) \approx_\varepsilon (U_\ell, \cdot).$$

Thus, except with probability $\sqrt{\varepsilon}$ over the fixings of $Y_2, \mathcal{A}(Y)_2$ it holds that $\mathsf{Raz}(X,Y_2)$ is $\sqrt{\varepsilon}$-close to uniform. As for the entropy loss of Y_3 incurred by these fixings,

$$\widetilde{H}_\infty\left(Y_3 \mid Y_2, \mathcal{A}(Y)_2\right) = \widetilde{H}_\infty\left(Y_2 \circ Y_3 \mid Y_2, \mathcal{A}(Y)_2\right) \geq 0.99(d_2+d_3) - 2d_2 \geq 0.8d_3,$$

and so by Lemma 4.9, except with probability ε over the fixings of $Y_2, \mathcal{A}(Y)_2$, it holds that Y_3 has min-entropy rate larger than 0.5.

To summarize, except with probability $O(\sqrt{\varepsilon})$ over all fixings done so far, we have that

- The joint distribution of $\mathsf{Raz}(X, Y_2)$, $\mathsf{Raz}(X, \mathcal{A}(Y)_2)$ is independent of the joint distribution of $Y_3, \mathcal{A}(Y)_3$.
- The min-entropy of Y_3 is bounded below by

$$\frac{d_3}{2} \geq \frac{9d}{20} = \Omega\left(a \cdot \log\left(\frac{am}{\varepsilon}\right)\right) = \Omega\left(a \cdot \log\left(\frac{a\ell}{\varepsilon}\right)\right), \tag{6}$$

  where we used the lemma hypothesis on d for the second inequality and that $d \geq 2d_1$ for the first inequality. The last equality follows by our choice of ℓ.
- $\mathsf{Raz}(X, Y_2)$ is $O(\sqrt{\varepsilon})$-close to uniform.

Therefore, we can apply Theorem 4.12 and conclude that

$$(\mathsf{NMExt}(X, Y), \mathsf{NMExt}(X, \mathcal{A}(Y)), Y) \approx_{O(\sqrt{\varepsilon})} (U_m, \cdot).$$

Note that the hypothesis of Theorem 4.12 holds. In particular, Equation (3) holds by our choice of ℓ, and Equation (4) follows by Equation (6). This concludes the proof. ◀

6 Reducing the Entropy Requirement of Non-Malleable Extractors

In this section we prove the following lemma which is a formal restatement of Lemma 2.3.

▶ **Lemma 6.1.** *There exists a universal constant $\alpha > 0$ such that the following holds. Let $\mathsf{NMExt}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{\log(1/\varepsilon)}$ be an explicit non-malleable extractor with error ε for entropy k. Let m be any integer. Assume that*

$$\begin{aligned} k &= \Omega(d^\alpha \cdot \log(n/\varepsilon)), \\ d &= \Omega\left(\log^4(1/\varepsilon) \cdot \log^2 m\right), \\ m &= O\left(\sqrt{k}/\log(1/\varepsilon)\right). \end{aligned}$$

Then, there exists an explicit non-malleable extractor $\mathsf{NMExt}'\colon \{0,1\}^n \times \{0,1\}^{d'} \to \{0,1\}^m$ for entropy $k' = k/d^\alpha$ with seed length $d' = O(d)$ and error $O(\varepsilon^{1/4})$.

The proof of Lemma 6.1 consists of two steps. First, we show how to construct an advice generator for entropy k' given a non-malleable extractor for entropy $k > k'$. This is done in the next subsection. Then, we apply Lemma 5.3 to obtain a non-malleable extractor for entropy k' using this advice generator. This second step is covered in Section 6.2.

6.1 From non-malleable extractors to advice generators for lower entropy

In this section we prove the following lemma.

▶ **Lemma 6.2.** *There exists a universal constant $c > 1$ such that the following holds. Let $\mathsf{NMExt}\colon \{0,1\}^n \times \{0,1\}^{d_1} \to \{0,1\}^{\log(1/\varepsilon)}$ be an explicit non-malleable extractor for entropy k with error ε. Let $\delta > 0$, and set $\Delta = \delta^{-c}$. Assume that*

$$\delta k = \Omega((\Delta + \log d_1) \cdot \log(1/\varepsilon)). \tag{7}$$

Then, there exists an explicit d_1-nice advice generator $\mathsf{AdvGen}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^a$ for entropy δk with error $O(\sqrt{\varepsilon}) + 2^{-\Omega(\delta^2 n)}$, seed length $d = O(d_1)$, and $a = O(\Delta \cdot \log(1/\varepsilon))$ output bits.

Proof. Let $d_2 = 1000d_1$ and set $d = d_1 + d_2$. For the construction of AdvGen we make use of the following building blocks:

- Let $\{\mathsf{Cond}_i \colon \{0,1\}^n \to \{0,1\}^u\}_{i=1}^{\Delta}$ be the sequence of efficiently computable functions given by Theorem 4.5 when applied with n and δ. By Theorem 4.5, $u = n \cdot \mathrm{poly}(\delta)$ and $\Delta = \delta^{-c}$ for some universal constant c. This constant will be the constant c introduced in the statement of the lemma.
- Let $\mathsf{ECC} \colon \{0,1\}^{d_2} \to \colon \{0,1\}^{D_2}$ be the error correcting code from Theorem 4.4 set with relative distance $1/4$. By Theorem 4.4, $D_2 = O(d_2)$.
- Let $r = \log_{4/3}(1/\varepsilon)$ and set $m = r \cdot \log_2 D_2$. Let $\mathsf{Ext} \colon \{0,1\}^n \times \{0,1\}^{d_1} \to \{0,1\}^m$ be the extractor from Theorem 4.2. Note that we use a seed of the same length d_1 as was used for the non-malleable extractor NMExt. This suffices for us since by Theorem 4.2, a seed of that length is sufficient for Ext to have error ε. By identifying $\{0,1\}^m$ with $[D_2]^r$, we interpret the output of Ext as an r-tuple in $[D_2]$.

Set $a = \Delta \cdot \log_2(1/\varepsilon) + \log_{4/3}(1/\varepsilon)$. We define the function $\mathsf{AdvGen} \colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^a$ as follows:

$$\mathsf{AdvGen}(x,y) = \mathsf{NMExt}(\mathsf{Cond}_1(x), y_1) \circ \cdots \circ \mathsf{NMExt}(\mathsf{Cond}_\Delta(x), y_1) \circ \mathsf{ECC}(y_2)_{\mathsf{Ext}(x,y_1)}.$$

Note that we feed as a first argument to NMExt u bit strings rather than the n bit strings it expects. We do so for simplicity of presentation. This minor technical issue can be overcome by appending zeros to the u bit string so to obtain an n bit string, and instructing the extractor to ignore these zeros.

Having the definition of AdvGen at hand, we turn to the analysis. Let X be an $(n, \delta k)$-source and let Y be an independent random variable that is uniformly distributed over $\{0,1\}^d$. Consider a function $\mathcal{A} \colon \{0,1\}^d \to \{0,1\}^d$ with no fixed points. By Theorem 4.5 (and ignoring the convexity, for ease of readability) there exists $g \in [\Delta]$ such that $\mathsf{Cond}_g(X)$ is $2^{-\Omega(\delta^2 n)}$-close to having min-entropy k. Therefore, by Lemma 4.14, there exists a set $B \subset \{0,1\}^{d_1}$ of density $\sqrt{\varepsilon}$ such that for any $y_1 \notin B$ and any d_1-bit string $y_1' \neq y_1$, it holds that

$$(\mathsf{NMExt}(\mathsf{Cond}_g(X), y_1), \mathsf{NMExt}(\mathsf{Cond}_g(X), y_1')) \approx_{\sqrt{\varepsilon}+2^{-\Omega(\delta^2 n)}} (U, \cdot)\,. \tag{8}$$

We now fix $y_1 \sim Y_1$ and $y_1' \sim \mathcal{A}(Y)_1$. Clearly, these fixings do not introduce dependencies between X, Y. Furthermore, by the above, we can aggregate $\sqrt{\varepsilon}$ to the total error and assume that $y_1 \notin B$. We continue by considering two cases.

Case 1 – $y_1 \neq y_1'$. As the output length of NMExt is $\log(1/\varepsilon)$, Equation (8) implies that the probability that $\mathsf{NMExt}(\mathsf{Cond}_g(X), y_1) = \mathsf{NMExt}(\mathsf{Cond}_g(X), y_1')$ is bounded above by $O(\sqrt{\varepsilon}) + 2^{-\Omega(\delta^2 n)}$. Thus, in this case, except with probability $O(\sqrt{\varepsilon}) + 2^{-\Omega(\delta^2 n)}$ we have that $\mathsf{AdvGen}(X,Y) \neq \mathsf{AdvGen}(X, \mathcal{A}(Y))$.

Case 2 – $y_1 = y_1'$. This case follows the same idea as in the proofs of Theorem 8.1 and Lemma 7.1. Conditioned on $y_1 = y_1'$ we have that $Y_2 \neq \mathcal{A}(Y)_2$. Hence, the codewords $\mathsf{ECC}(Y_2), \mathsf{ECC}(\mathcal{A}(Y)_2)$ agree on at most $3/4$ of the coordinates of $[D_2]$. Hence, the set of r-tuples over $[D_2]$ for which $\mathsf{ECC}(Y_2)$ agrees with $\mathsf{ECC}(\mathcal{A}(Y)_2)$ on all r coordinates of the tuple has density at most $(3/4)^r = \varepsilon$ within $[D_2]^r$. We denote this set of r-tuples by $B' \subseteq [D_2]^r$.

Recall that Ext is a strong seeded extractor with ε. Thus, except for probability $\sqrt{\varepsilon}$ over the choice of y_1, we have that $\mathsf{Ext}(X, y_1)$ is $\sqrt{\varepsilon}$-close to uniform. Therefore, for such y_1,

$\mathbf{Pr}[\mathsf{Ext}(X, y_1) \in B'] = O(\sqrt{\varepsilon})$. Hence, except with probability $O(\sqrt{\varepsilon})$ over the fixings done so far, we have that also in this case, $\mathsf{AdvGen}(X, Y) \neq \mathsf{AdvGen}(X, \mathcal{A}(Y))$.

As for niceness property, by Lemma 4.8, the fixings of $Y_1, \mathcal{A}(Y)_1$ reduce the average min-entropy of Y_2 by a most $2d_1$. Once $Y_1, \mathcal{A}(Y)_1$ are fixed, we have that $\mathsf{Ext}(X, Y_1)$ and $\mathsf{Ext}(X, \mathcal{A}(Y)_1)$ are deterministic functions of X. Thus, we can fix the latter random variables without introducing dependencies between X, Y. Further, by Lemma 4.8, the average min-entropy of X decreases by at most $2m$. After these fixings, $\mathsf{ECC}(Y_2)_{\mathsf{Ext}(X,Y_1)}$ and $\mathsf{ECC}(\mathcal{A}(Y)_2)_{\mathsf{Ext}(X,\mathcal{A}(Y)_1)}$ are deterministic functions of Y that consist of r bits each. Thus, fixing these random variables will reduce the average min-entropy of Y by at most $2r$. Further, these fixings do not introduce any dependencies between X, Y.

Finally, after all of the fixings done so far, $\mathsf{AdvGen}(X, Y)$ and $\mathsf{AdvGen}(X, \mathcal{A}(Y))$ are deterministic functions of X. We can therefore fix these random variables, which will result in an entropy-loss of at most $2\Delta \cdot \log(1/\varepsilon)$. Again, these last fixings do not introduce any dependencies between X, Y.

To summarize, in the process of fixing $\mathsf{AdvGen}(X, Y)$, $\mathsf{AdvGen}(X, \mathcal{A}(Y))$, the random variable Y lost an average entropy of $2d_1 + 2\log_{4/3}(1/\varepsilon)$. Thus, by the choice of d_2, except with probability ε over these fixings, Y_2 has min-entropy rate 0.99. As for X, the fixings reduced its average min-entropy by

$$2\Delta \log(1/\varepsilon) + 2m = O\left(\Delta \log(1/\varepsilon) + \log(d)\log(1/\varepsilon)\right),$$

and so by Equation (7), except with probability ε over the fixings of $\mathsf{AdvGen}(X, Y)$, $\mathsf{AdvGen}(X, \mathcal{A}(Y))$, the source X has min-entropy rate 0.99. This concludes the proof. ◀

6.2 Proof of Lemma 6.1

Proof of Lemma 6.1. Let c be the constant from Lemma 6.2. Set $\alpha = 1/(4c)$ and set $\delta = d^{-\alpha}$. We borrow the notation from Lemma 6.2 and write $\Delta = \delta^{-c} = d^{1/4}$. First, we apply Lemma 6.2 with the non-malleable extractor NMExt and δ as set above. To show that this application is valid one needs to verify that

▶ **Claim 6.3.** $\delta k = \Omega((\Delta + \log d) \cdot \log(1/\varepsilon))$.

Proof. Note that $\Delta = d^{1/4} = \Omega(\log d)$. Thus, to prove the claim it suffices to show that $\delta k = \Omega(\Delta \cdot \log(1/\varepsilon))$. To verify that this inequality holds, it suffices to show that $k = \Omega(\Delta^2 \cdot \log(1/\varepsilon))$, or equivalently that $k = \Omega(\sqrt{d} \cdot \log(1/\varepsilon))$, which indeed follows by our assumption. ◀

Lemma 6.2 transforms the given NMExt to an efficiently computable d-nice advice generator $\mathsf{AdvGen}\colon \{0,1\}^n \times \{0,1\}^{O(d)} \to \{0,1\}^a$ for entropy δk with advice length $a = O(\Delta \cdot \log(1/\varepsilon))$ and error $O(\sqrt{\varepsilon}) + 2^{-\Omega(\delta^2 n)} = O(\sqrt{\varepsilon})$. Next, we would like to apply Lemma 5.3 to AdvGen so to obtain a non-malleable extractor $\mathsf{NMExt}'\colon \{0,1\}^n \times \{0,1\}^{O(d)} \to \{0,1\}^m$ for entropy δk, with error $O(\varepsilon^{1/4})$. To this end, we need to verify that the hypothesis of the lemma holds, which is guaranteed by the following cliam.

▶ **Claim 6.4.**

$$\begin{aligned} m &= O(\delta k/a), \\ d &= \Omega\left(a \cdot \log\left(\frac{am}{\varepsilon}\right) + \log(n/\varepsilon)\right), \\ \delta k &= \Omega\left(a \cdot \log\left(\frac{ad}{\varepsilon}\right) + \log(n/\varepsilon)\right), \end{aligned}$$

Proof. To prove the first inequality it suffices to show that $m = O(k/(\Delta^2 \cdot \log(1/\varepsilon)))$. Since $\Delta = d^{1/4}$ and $k > d$, it suffices to show that $m = O(\sqrt{k}/\log(1/\varepsilon))$, which follows by the hypothesis of the lemma. As for the second inequality, first note that $d = \Omega(\log(n/\varepsilon))$ as d is a seed for the non-malleable extractor NMExt. Thus, it suffices to verify that $d = \Omega(a \cdot \log(am/\varepsilon))$. Since $a = O(\Delta \cdot \log(1/\varepsilon))$, this inequality holds as long as $d = \Omega(\Delta^2 \cdot \log^2(1/\varepsilon) \cdot \log m)$. Since $\Delta^2 = \sqrt{d}$, it suffices to verify that $d = \Omega(\log^4(1/\varepsilon) \cdot \log^2 m)$, which holds by assumption.

As for the last inequality, we first show that $\delta k = \Omega(a \cdot \log(ad/\varepsilon))$ and afterwards turn to verify that $\delta k \geq \log(n/\varepsilon)$. For the first inequality, it suffices to show that $k = \Omega(\Delta^2 \cdot \log^2(1/\varepsilon) \cdot \log d)$. As $\Delta^2 = \sqrt{d}$ and since $\sqrt{d} = \Omega(\log^2(1/\varepsilon))$, it suffices to show that $k = \Omega(d \cdot \log d)$ which follows by assumption. Further, as $\delta = d^{-\alpha}$, the inequality $\delta k \geq \log(n/\varepsilon)$ follows. ◀

By the above claim, NMExt' is indeed a non-malleable extractor for min-entropy δk, with seed length $O(d)$, error $O(\varepsilon^{1/4})$ and m output bits. This concludes the proof. ◀

7 Increasing the Output Length of a Non-Malleable Extractor

A general tool we use is an algorithm that increases the output length of a given non-malleable extractor in a black-box manner. This is given by the following lemma which is a formal restatement of Lemma 2.4.

▶ **Lemma 7.1.** *There exists a universal constant $\alpha > 0$ such that the following holds. Let $\mathsf{NMExt}\colon \{0,1\}^n \times \{0,1\}^{d_1} \to \{0,1\}^{\log(1/\varepsilon)}$ be an explicit non-malleable extractor for entropy k with error ε such that*

$$k = \Omega\left(\log(d_1/\varepsilon) \cdot \log(1/\varepsilon) + \log(n/\varepsilon)\right).$$

Then, for any $m < \alpha k/\log(1/\varepsilon)$ there exists an explicit non-malleable extractor $\mathsf{NMExt}'\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ for entropy k with error $O(\varepsilon^{1/4})$, having seed length

$$d = O\left(d_1 + \log(m/\varepsilon) \cdot \log(1/\varepsilon)\right).$$

Note that Lemma 2.4 follows by Lemma 7.1 for constant ε by setting $m = \Omega(k)$. Indeed, the expression $\log(m/\varepsilon) \cdot \log(1/\varepsilon)$ in the resulted seed length d is $O(\log k)$ which is always smaller than d_1 (as the seed length of any seeded extractor, in particular NMExt, is at least $\log(n-k)$), and so in the setting of Lemma 2.4, $d = O(d_1)$.

Proof of Lemma 7.1. During the proof we make use of the following notation. Given a string $y \in \{0,1\}^d$, we write $y = y_1 \circ y_2$ where $|y_1| = d_1$ and define $d_2 = d - d_1 = 500d_1$. We make use of the following building blocks:

- Let $\mathsf{ECC}\colon \{0,1\}^{d_2} \to \colon \{0,1\}^{D_2}$ be the error correcting code from Theorem 4.4 set with relative distance $1/4$. By Theorem 4.4, $D_2 = O(d_2)$.
- Let $r = \log_{4/3}(1/\varepsilon)$ and set $v = r \cdot \log_2 D_2$. Let $\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^{d_1} \to \{0,1\}^v$ be the extractor from Theorem 4.2. Note that we use a seed of the same length d_1 as was used for the non-malleable extractor NMExt. By identifying $\{0,1\}^v$ with $[D_2]^r$, we interpret the output of Ext as an r-tuple over $[D_2]$.

We proceed by proving the following claim.

▶ **Claim 7.2.** *The function*

$$\mathsf{AdvGen}(x,y) = \mathsf{NMExt}(x,y_1) \circ \mathsf{ECC}(y_2)_{\mathsf{Ext}(x,y_1)}$$

is a d_1-nice advice generator for entropy k with error $O(\sqrt{\varepsilon})$.

Proof. By Lemma 4.14, there exists a set $B \subset \{0,1\}^{d_1}$ of density $\sqrt{\varepsilon}$ such that for any $y_1 \notin B$ and any d_1-bit string $y_1' \neq y_1$, it holds that

$$(\mathsf{NMExt}(X,y_1), \mathsf{NMExt}(X,y_1')) \approx_{\sqrt{\varepsilon}} (U,\cdot)\,. \tag{9}$$

We now fix $y_1 \sim Y_1$ and $y_1' \sim \mathcal{A}(Y)_1$. Clearly, these fixings do not introduce dependencies between X,Y. Furthermore, by the above, we can aggregate $\sqrt{\varepsilon}$ to the total error and assume that $y_1 \notin B$. We continue by considering two cases.

Case 1 – $y_1 \neq y_1'$. In this case Equation (9) holds. In particular, as NMExt has output length $\log(1/\varepsilon)$, the probability that $\mathsf{NMExt}(X,y_1) = \mathsf{NMExt}(X,y_1')$ is bounded above by $O(\sqrt{\varepsilon})$. Thus, in this case, except with probability $O(\sqrt{\varepsilon})$ we have that $\mathsf{AdvGen}(X,Y) \neq \mathsf{AdvGen}(X,\mathcal{A}(Y))$.

Case 2 – $y_1 = y_1'$. Here we follow the idea of [4] from Theorem 8.1. Conditioned on $y_1 = y_1'$ we have that $Y_2 \neq \mathcal{A}(Y)_2$, and so the codewords $\mathsf{ECC}(Y_2), \mathsf{ECC}(\mathcal{A}(Y)_2)$ agree on at most $3/4$ of the coordinates of $[D_2]$. Hence, the set of r-tuples over $[D_2]$ for which $\mathsf{ECC}(Y_2)$ agrees with $\mathsf{ECC}(\mathcal{A}(Y)_2)$ on all r coordinates of the tuple has density at most $(3/4)^r = \varepsilon$ within $[D_2]^r$. We denote this set of r-tuples by $B' \subseteq [D_2]^r$.

Recall that Ext is a strong seeded extractor with error ε. Moreover, $H_\infty(X) \geq k = \Omega(v)$, and so except for probability $\sqrt{\varepsilon}$ over the choice of y_1, we have that $\mathsf{Ext}(X,y_1)$ is $\sqrt{\varepsilon}$-close to uniform. For any such y_1 we have that $\mathbf{Pr}[\mathsf{Ext}(X,y_1) \in B'] = O(\sqrt{\varepsilon})$. Thus, except with probability $O(\sqrt{\varepsilon})$ over the fixings done so far, we have that also in this case $\mathsf{AdvGen}(X,Y) \neq \mathsf{AdvGen}(X,\mathcal{A}(Y))$.

As for the niceness property, by Lemma 4.8, the fixings of $Y_1, \mathcal{A}(Y)_1$ reduce the average min-entropy of Y_2 by a most $2d_1$. Once $Y_1, \mathcal{A}(Y)_1$ are fixed, we have that $\mathsf{Ext}(X,Y_1)$, $\mathsf{Ext}(X,\mathcal{A}(Y)_1)$, $\mathsf{NMExt}(X,Y_1)$, and $\mathsf{NMExt}(X,\mathcal{A}(Y)_1)$ are all deterministic functions of X. Thus, we can fix the latter random variables without introducing dependencies between X,Y. Further, by Lemma 4.8, the average min-entropy of X decreases by at most $2v + 2\log(1/\varepsilon) = O(\log(1/\varepsilon)\cdot\log d_1)$.

After these fixings, $\mathsf{ECC}(Y_2)_{\mathsf{Ext}(X,Y_1)}$ and $\mathsf{ECC}(\mathcal{A}(Y)_2)_{\mathsf{Ext}(X,\mathcal{A}(Y)_1)}$ are deterministic functions of Y, each consists of r bits. Thus, fixing these random variables will reduce the average min-entropy of Y by at most $2r = O(\log(1/\varepsilon))$. Further, these fixings do not introduce any dependencies between X,Y. Note that after all of the fixings done so far, $\mathsf{AdvGen}(X,Y)$ and $\mathsf{AdvGen}(X,\mathcal{A}(Y))$ are fixed.

To summarize, in the process of fixing $\mathsf{AdvGen}(X,Y)$, $\mathsf{AdvGen}(X,\mathcal{A}(Y))$, the random variable Y_2 lost an average entropy of $2d_1 + 2r$. Since $d_2 = 500d_1$ we have that except with probability ε over these fixings, Y_2 has min-entropy rate 0.99. As for X, the fixings reduced its average min-entropy by $O(\log(1/\varepsilon)\cdot\log d_1)$, and so except with probability ε, X has min-entropy rate 0.99 conditioned on these fixings. ◀

To conclude the proof we apply Lemma 5.3 with AdvGen defined above and the parameter m. The hypothesis of Lemma 5.3 is met due to our hypothesis on m,d,k and since $a = O(\log(1/\varepsilon))$. ◀

8 Proof of Theorem 2.1

In this section we prove Theorem 2.1. For the proof we make use of the advice generator of [4] that is given by the following theorem.

▶ **Theorem 8.1** ([4]). *For any integer n and all $\varepsilon > 0$ there exists a $O(\log(n/\varepsilon))$-nice advice generator* $\mathsf{AdvGen}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^a$ *for entropy*

$$k = \Omega\left(\log(1/\varepsilon) \cdot \log\log(n/\varepsilon)\right) \tag{10}$$

with error ε, seed length $d = O(\log(n/\varepsilon))$, and $a = O(\log(n/\varepsilon))$ output bits.

We defer the proof of Theorem 8.1 to Section 8.1 and start by proving Theorem 2.1.

Proof of Theorem 2.1. Our first step is to apply Lemma 5.3 to the advice generator AdvGen given by Theorem 8.1. For simplicity, we consider a constant error ε. One can easily verify that this gives us a non-malleable extractor NMExt_0 for entropy $\Omega(\log n \cdot \log\log n)$ having seed length $O(\log n \cdot \log\log n)$.

Our second step would be to reduce the entropy requirement to $\Omega(\log n)$. To this end, we apply Lemma 6.2 to NMExt_0 with $\delta = O(1/\log\log n)$. One can easily verify that the hypothesis of Lemma 6.2 holds with this choice of δ. Thus, we obtain an advice generator with seed length $O(\log n \cdot \log\log n)$ and advice length $a = \operatorname{poly}\log\log n$ for entropy $\Omega(\log n)$.

We now apply Lemma 5.3 to AdvGen with constant output length m so to obtain a non-malleable extractor NMExt_1 with seed length $O(\log n \cdot \log\log n)$ for entropy $\Omega(\log n)$. One can easily verify that the hypothesis of Lemma 5.3 are met by our choice of δ.

To obtain our final non-malleable extractor, denoted by NMExt_2, we apply Lemma 7.1 to NMExt_1 with $m = \Omega(\log n)$. One can again easily verify that all the conditions of Lemma 7.1 hold and that the resulting non-malleable extractor, NMExt_2, supports entropy $\Omega(\log n)$, has seed length $O(\log n \cdot \log\log n)$, and has output length $\Omega(\log n)$. ◀

8.1 The advice generator of [4]

In this section we prove Theorem 8.1. We give a full proof here since the theorem as stated is somewhat implicit in [4]. Nevertheless, we stress that all the ideas already appear in [4].

Proof of Theorem 8.1. We start by describing the construction of AdvGen and then turn to the analysis. Let c be the universal constant from Theorem 4.2. Split $y = y_1 \circ y_2$, where y_1 consists of $d_1 = c \cdot \log(n/\varepsilon)$ bits, and set $d_2 = d - d_1$, where $d = c' \cdot d_1$ for some large enough constant c'. We define $\mathsf{AdvGen}(x, y) = y_1 \circ \phi(x, y)$, where $\phi(x, y)$ is described next. For the definition of ϕ we make use of the following building blocks:

- Let $\mathsf{ECC}\colon \{0,1\}^{d_2} \to \colon \{0,1\}^{D_2}$ be the error correcting code from Theorem 4.4 set with relative distance $1/4$. By Theorem 4.4, $D_2 = O(d_2)$.
- Let $r = \log_{4/3}(1/\varepsilon)$ and set $m = r \cdot \log_2 D_2$. Let $\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^{d_1} \to \{0,1\}^m$ be the extractor from Theorem 4.2, set with error ε. Recall that we set d_1 to be large enough as required by a seed for Ext. Moreover, one can verify that by our assumption on k given by Equation (10), $k \geq 2m$.

Let $z = \mathsf{Ext}(x, y_1)$. We identify $\{0,1\}^m$ with $[D_2]^r$ and let $i_1(z), \ldots, i_r(z)$ be elements in $[D_2]$ corresponding to the r consecutive length $\log_2 D_2$ substrings of z. For $j = 1, \ldots, r$, we define

$$\phi(x, y)_j = \mathsf{ECC}(y_2)_{i_j(z)}.$$

We now turn to the analysis. First, note that the output length of AdvGen is $a = O(\log(n/\varepsilon))$ as stated. Indeed, the output is a concatenation of y_1 with $\phi(x,y)$, where $|y_1| = d_1 = O(\log(n/\varepsilon))$ and $\phi(x,y)$ consists of $r = O(\log(1/\varepsilon))$ bits. Now, by the strongness of Ext we have that

$$(\mathsf{Ext}(X,Y_1),Y_1) \approx_\varepsilon (U_m,\cdot)\,.$$

Further, by Lemma 4.10

$$(\mathsf{Ext}(X,Y_1),Y_1,\mathcal{A}(Y)_1) \approx_\varepsilon (U_m,\cdot)\,,$$

Indeed, the hypothesis of Lemma 4.10 is met as conditioned on any fixing of Y_1, the random variable $\mathsf{Ext}(X,Y_1)$ is independent of $\mathcal{A}(Y)_1$. Furthermore, by Lemma 4.8,

$$\widetilde{H}_\infty\left(Y_2 \mid Y_1,\mathcal{A}(Y)_1\right) \geq d_2 - 2d_1.$$

Thus, except with probability $2\sqrt{\varepsilon}$ over the fixing of $y_1 \sim Y_1$, $y_1' \sim \mathcal{A}(Y)_1$, it holds that $\mathsf{Ext}(X,y_1)$ is $\sqrt{\varepsilon}$-close to uniform and that

$$H_\infty(Y_2) \geq d_2 - 2d_1 - \log(1/\varepsilon) \geq 0.999d_2. \tag{11}$$

From this point on we condition on a fixings of y_1, y_1' for which $\mathsf{Ext}(X,y_1)$ is $\sqrt{\varepsilon}$-close to uniform and for which Equation (11) holds, and aggregate $2\sqrt{\varepsilon}$ to the total error.

As y_1 is a prefix of $\mathsf{AdvGen}(x,y)$ and y_1' is a prefix of $\mathsf{AdvGen}(x,\mathcal{A}(y))$, we have that if $y_1 \neq y_1'$ then $\mathsf{AdvGen}(X,Y) \neq \mathsf{AdvGen}(X,\mathcal{A}(Y))$. Therefore, we may assume that $y_1 = y_1'$. Since $\mathcal{A}$ has no fixed points it holds that $Y_2 \neq \mathcal{A}(Y)_2$. Therefore, the codewords $\mathsf{ECC}(Y_2),\mathsf{ECC}(\mathcal{A}(Y)_2)$ agree on at most $3/4$ fraction of the coordinates of $[D_2]$, and so the set of r-tuples over $[D_2]$ in which $\mathsf{ECC}(Y_2)$ equals $\mathsf{ECC}(\mathcal{A}(Y)_2)$ in all r coordinates has density at most $(3/4)^r = \varepsilon$ within $[D_2]^r$. We denote this set of "bad" r-tuples by $B \subseteq [D_2]^r$.

As $\mathsf{Ext}(X,y_1)$ is $\sqrt{\varepsilon}$-close to uniform, we have that

$$\mathbf{Pr}[\mathsf{Ext}(X,y_1) \in B] \leq \varepsilon + \sqrt{\varepsilon} \leq 2\sqrt{\varepsilon},$$

and so

$$\mathop{\mathbf{Pr}}_{\substack{x\sim X\\ y\sim Y}}[\mathsf{AdvGen}(x,y) = \mathsf{AdvGen}(x,\mathcal{A}(y))] = O(\sqrt{\varepsilon}).$$

As for the niceness property, note that conditioned on the fixings done so far, namely, the fixings of Y_1 and $\mathcal{A}(Y)_1$ it holds that both $\mathsf{Ext}(X,Y_1)$, $\mathsf{Ext}(X,\mathcal{A}(Y)_1)$ are deterministic functions of X. As these random variables consist of $2m$ bits altogether, we have that conditioned on the further fixings of $\mathsf{Ext}(X,Y_1)$, $\mathsf{Ext}(X,\mathcal{A}(Y)_1)$, the average min-entropy of X is bounded below by $k - 2m$. Hence, by Lemma 4.9, except with probability ε over the further fixings of these random variables,

$$H_\infty(X) \geq k - 2m - \log(1/\varepsilon) \geq 0.99k.$$

Note that the fixing of $\mathsf{Ext}(X,Y_1)$, $\mathsf{Ext}(X,\mathcal{A}(Y)_1)$ does not reduce the entropy of Y_2 and does not introduce any correlation between X,Y.

Conditioned on the fixings done so far, we have that $\mathsf{Ext}(X,Y_1)$, $\mathsf{Ext}(X,\mathcal{A}(Y)_1)$ are fixed, and so $\phi(X,Y)$, $\phi(X,\mathcal{A}(Y))$ are deterministic functions Y that consist of $2r$ bits. Thus, we can further condition the fixings of $\phi(X,Y)$, $\phi(X,\mathcal{A}(Y))$, which results in the fixings of $\mathsf{AdvGen}(X,Y)$ and $\mathsf{AdvGen}(X,\mathcal{A}(Y))$. Furthermore, as $2r + \log(1/\varepsilon) \leq 0.009d_2$, conditioned on these fixings we have that Y_2 has min-entropy rate 0.99 except with probability ε. Note that these fixings do not introduce dependencies between X,Y. Further, note that the total error incurred so far can be reduced from $O(\sqrt{\varepsilon})$ to ε without need for any change in the hypothesis of the theorem. This concludes the proof of the theorem. ◀

9 Proof of Theorem 2.2

Building on results developed so far, in this section we prove Theorem 2.1.

Proof of Theorem 2.2. Set $m = \log n$. Our starting point is the explicit non-malleable extractor $\mathsf{NMExt}_0 \colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ from Theorem 4.6 that supports entropy $0.6n$, has error $1/\log n$, and seed length $d = O(\log n)$. We apply Lemma 6.1 to NMExt_0 with m as set above so to obtain a second non-malleable extractor $\mathsf{NMExt}_1 \colon \{0,1\}^n \times \{0,1\}^{d_1} \to \{0,1\}^m$, where $d_1 = O(d)$. One can easily verify that the hypothesis of Lemma 6.1 holds, and so Lemma 6.1 guarantees that NMExt_1 is a non-malleable extractor for entropy $k_1 = k_0/d_0^\alpha = \Omega(n/(\log n)^\alpha)$, where α is the universal constant from Lemma 6.1. By Lemma 6.1, the error of NMExt_1 is $\varepsilon_1 = O((\log n)^{-1/4})$.

We apply Lemma 6.1 again, now to NMExt_1 with m as before. One can verify that the hypothesis of this application of Lemma 6.1 holds as well, and so we obtain a third non-malleable extractor $\mathsf{NMExt}_2 \colon \{0,1\}^n \times \{0,1\}^{d_2} \to \{0,1\}^m$, where $d_2 = O(d_1) = O(d)$, for min-entropy $k_2 = k_1/d_1^\alpha = \Omega(n/(\log n)^{2\alpha})$. The error of NMExt_2 is $\varepsilon_2 = O((\log n)^{-1/4^2})$

We repeat this process, producing a sequence of non-malleable extractors, always with m output bits. After r iterations we obtain a non-malleable extractor $\mathsf{NMExt}_r \colon \{0,1\}^n \times \{0,1\}^{d_r} \to \{0,1\}^m$, where $d_r = 2^{O(r)} \cdot \log n$, having error $\varepsilon_r = (\log n)^{-1/4^r}$ for entropy $k_r = \Omega\left(n/(\log n)^{\alpha r}\right)$. One can prove by induction that indeed for any constant r, these sequence of applications of Lemma 6.1 is valid. Notice that for any constant r the error is bounded above by ε – the desired constant error guarantee, assuming n is large enough. Thus, by setting $r = c/\alpha$, we obtain a non-malleable extractor with seed length $O(\log n)$ for entropy $\Omega(n/\log^c n)$ with error ε.

Lastly, we increase the output length of NMExt_r by applying Lemma 7.1 with $m = \Omega(k)$ to NMExt_r so to obtain our final non-malleable extractor $\mathsf{NMExt}' \colon \{0,1\}^n \times \{0,1\}^{d'} \to \{0,1\}^m$. One can easily verify that the hypothesis of Lemma 7.1 is met and that $d' = O(\log n)$. ◀

10 From Non-Malleable Extractors to t-Non-Malleable Extractors

In this section we prove the following lemma, which is a formal restatement of Lemma 2.5.

▶ **Lemma 10.1.** *Let $t \geq 1$ be an integer. Let $\mathsf{NMExt} \colon \{0,1\}^n \times \{0,1\}^{d_1} \to \{0,1\}^{\log(1/\varepsilon)}$ be an explicit non-malleable extractor for entropy k with error ε such that*

$$k = \Omega\left(t \cdot \log(td_1/\varepsilon) \cdot \log(1/\varepsilon) + \log(n/\varepsilon)\right).$$

Then, for any $m = O(k/(t \cdot \log(1/\varepsilon)))$ there exists an explicit t-non-malleable extractor $\mathsf{NMExt}' \colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ for entropy k with error $O(t \cdot \varepsilon^{1/4})$, having seed length

$$d = O\left(t^2 d_1 + t \cdot \log(tm/\varepsilon) \cdot \log(1/\varepsilon)\right).$$

The proof of Lemma 10.1 builds on what we call t-advice generators that generalize Definition 5.1.

▶ **Definition 10.2** (t-advice generators)**.** For an integer $t \geq 1$, a function $\mathsf{AdvGen} \colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^a$ is called a *t-advice generator* for entropy k with error ε if the following holds. For any (n,k)-source X, an independent random variable Y that is uniform over $\{0,1\}^d$, and any functions $\{\mathcal{A}_i \colon \{0,1\}^d \to \{0,1\}^d\}_{i=1}^t$ with no fixed points, it holds that

$$\Pr_{\substack{x \sim X \\ y \sim Y}}\left[\exists i \in [t] \quad \mathsf{AdvGen}(x,y) = \mathsf{AdvGen}(x, \mathcal{A}_i(y))\right] \leq \varepsilon.$$

Note that a 1-advice generator is an advice generator as defined in Definition 5.1. Similarly to our reduction in Lemma 5.3, some extra guarantee from the t-advice generator is needed for the reduction from t-non-malleable extractors to t-advice generators. This is encapsulated in the following definition.

▶ **Definition 10.3** (Nice t-advice generators)**.** A t-advice generator $\mathsf{AdvGen}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^a$ for entropy k with error ε is said to be d_1-nice if the following holds. Let X be an (n,k)-source, let Y be a random variable that is independent of X and is uniformly distributed over $\{0,1\}^d$, and let $\{\mathcal{A}_i\colon \{0,1\}^d \to \{0,1\}^d\}_{i=1}^t$ be functions with no fixed points. Then, except with probability ε over the fixings of $\mathsf{AdvGen}(X,Y)$, $\{\mathsf{AdvGen}(X,\mathcal{A}_i(Y))\}_{i=1}^t$ it holds that:

- X, Y are independent.
- $H_\infty(X) \geq 0.99k$.
- The length $d - d_1$ suffix of Y has min-entropy rate $1 - 1/(100t)$.

Note that a nice 1-advice generator is a nice advice generator as defined in Section 5. Mimicking the proof of Lemma 5.3 we obtain the following lemma.

▶ **Lemma 10.4.** *There exist universal constants $0 < c < 1 < c', c''$ such that the following holds. Let $\mathsf{AdvGen}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^a$ be an explicit t-advice generator for entropy k with error ε that is d_1-nice, with $d_1 \leq d/2$. Then, for any integer m such that*

$$m \leq c \cdot k/(at)$$
$$d \geq c't \cdot \max\left(a \cdot \log\left(\frac{atm}{\varepsilon}\right), \log(n/\varepsilon)\right),$$
$$k \geq c'' \cdot \max\left(at \cdot \log\left(\frac{ad}{\varepsilon}\right), \log(n/\varepsilon)\right),$$

there exists a t-non-malleable extractor $\mathsf{NMExt}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ for entropy k with error $O(\sqrt{\varepsilon})$.

Proof. We start by describing the construction of NMExt and then turn to the analysis. Given a string $y \in \{0,1\}^d$, we partition y to three consecutive substrings $y = y_1 \circ y_2 \circ y_3$, where $|y_1| = d_1$, $|y_2| = d_2 = \Omega(\log(n/\varepsilon))$ is a sufficient length for a seed of the extractor from Theorem 4.3 set with error ε, and $|y_3| = d_3 = (10t-1)d_2$. By our hypothesis, d is large enough so to satisfy these properties. We make use of the following building blocks:

- Let $\mathsf{Raz}\colon \{0,1\}^n \times \{0,1\}^{d_2} \to \{0,1\}^\ell$ be the extractor from Theorem 4.3, where

  $$\ell = c''' \cdot \max(atm, at\log(ad/\varepsilon))$$

  for some suitable constant c''' to be chosen next. By our choice of d_2, the error of Raz is bounded above by ε.
- Let $\mathsf{AdvCB}\colon \{0,1\}^{d_3} \times \{0,1\}^\ell \times \{0,1\}^a \to \{0,1\}^m$ be the t-correlation breaker with advice from Theorem 4.12 set with error ε. By Theorem 4.12, c''' can be chosen such that the output length of AdvCB is indeed m.

With the notation set and using the building blocks above, we define

$$\mathsf{NMExt}(x,y) = \mathsf{AdvCB}\left(y_3, \mathsf{Raz}(x,y_2), \mathsf{AdvGen}(x,y)\right).$$

We now turn to the analysis. Let X be an (n,k)-source, let Y be an independent random variable that is uniformly distributed over $\{0,1\}^d$, and let $\{\mathcal{A}_i\colon \{0,1\}^d \to \{0,1\}^d\}_{i=1}^t$

be functions with no fixed points. As AdvGen is a d_1-nice t-advice generator with error ε, we have that except with probability ε over the fixings of $\alpha \sim \mathsf{AdvGen}(X,Y)$, $\{\alpha^i \sim \mathsf{AdvGen}(X,\mathcal{A}_i(Y))\}_{i=1}^t$, it holds that

- $\alpha \notin \{\alpha^1,\ldots,\alpha^t\}$.
- X,Y remain independent.
- $H_\infty(X) \geq 0.99k$.
- The length $d-d_1$ suffix of Y has min-entropy rate $1-1/(100t)$.

We condition on such fixings. Next, we argue that except with probability ε over $y_3 \sim Y_3$ it holds that $Y_2 \mid (Y_3 = y_3)$ has min-entropy rate at least 0.6. To see this, apply Lemma 4.8 to obtain

$$\widetilde{H}_\infty(Y_2 \circ Y_3 \mid Y_3) \geq H_\infty(Y_2 \circ Y_3) - |Y_3| \geq \left(1 - \frac{1}{100t}\right)(d_2+d_3) - d_3 = 0.9d_2.$$

Thus, by Lemma 4.9, except with probability ε over $y_3 \sim Y_3$ it holds that

$$H_\infty(Y_2 \mid Y_3 = y_3) = H_\infty(Y_2 \circ Y_3 \mid Y_3 = y_3) \geq 0.9d_2 - \log(1/\varepsilon) \geq 0.6d_2.$$

Therefore, except with probability ε over the fixing of Y_3, the min-entropy rate of Y_2 is bounded below by 0.6. For the remaining of the proof we assume that the min-entropy rate of Y_2 is at least 0.6, and aggregate an additional error of ε to the total error.

As $H_\infty(X) \geq 0.99k \geq 2\ell$, $H_\infty(X) = \Omega(d_2)$, and since Y_2 is a $(d_2, 0.6d_2)$-source with $d_2 = \Omega(\log(n/\varepsilon))$, Theorem 4.3 implies that

$$(\mathsf{Raz}(X,Y_2), Y_2) \approx_\varepsilon (U_\ell, Y_2)\,.$$

As $\mathsf{Raz}(X,Y_2)$ is independent of the joint distribution of $\{(\mathcal{A}_i(Y))_2\}_{i=1}^t$ conditioned on the fixing of Y_2, Lemma 4.10 implies that

$$\left(\mathsf{Raz}(X,Y_2), Y_2, \{(\mathcal{A}_i(Y))_2\}_{i=1}^t\right) \approx_\varepsilon (U_\ell, \cdot)\,.$$

Thus, except with probability $\sqrt{\varepsilon}$ over the fixings of $Y_2, \{(\mathcal{A}_i(Y))_2\}_{i=1}^t$ it holds that $\mathsf{Raz}(X,Y_2)$ is $\sqrt{\varepsilon}$-close to uniform. As for the entropy loss of Y_3 resulted by these fixings,

$$\widetilde{H}_\infty\left(Y_3 \mid Y_2, \{(\mathcal{A}_i(Y))_2\}_{i=1}^t\right) \geq \left(1 - \frac{1}{100t}\right)(d_2+d_3) - (t+1)d_2 \geq 0.8d_3$$

and so except with probability ε over $Y_2, \{(\mathcal{A}_i(Y))_2\}_{i=1}^t$ it holds that Y_3 has min-entropy rate larger than 0.5. To summarize, except with probability $O(\sqrt{\varepsilon})$ over all fixings done so far, we have that

- The joint distribution of $\mathsf{Raz}(X,Y_2)$, $\{\mathsf{Raz}(X,(\mathcal{A}_i(Y))_2)\}_{i=1}^t$ is independent of the joint distribution of $Y_3, \{(\mathcal{A}_i(Y))_3\}_{i=1}^t$.
- The min-entropy of Y_3 is bounded below by

$$\frac{d_3}{2} \geq \frac{9d}{20} = \Omega\left(at \cdot \log\left(\frac{atm}{\varepsilon}\right)\right) = \Omega\left(at \cdot \log\left(\frac{a\ell}{\varepsilon}\right)\right), \tag{12}$$

  where we used the hypothesis on d and the choice of m for the second inequality and that $d \geq 2d_1$ for the first inequality. For the last inequality we used our choice of ℓ.
- $\mathsf{Raz}(X,Y_2)$ is $O(\sqrt{\varepsilon})$-close to uniform.

Therefore, we can apply Theorem 4.12 and conclude that

$$\left(\mathsf{NMExt}(X,Y), \{\mathsf{NMExt}(X,\mathcal{A}_i(Y))\}_{i=1}^t, Y\right) \approx_{O(\sqrt{\varepsilon})} (U_m, \cdot).$$

Note that indeed the hypothesis of Theorem 4.12 holds. In particular, Equation (3) holds by our choice of ℓ, and Equation (4) follows by Equation (12). This concludes the proof. ◀

We are now ready to prove Lemma 10.1

Proof of Lemma 10.1. Write $d = d_1 + d_2$, where $d_2 = 500t^2 d_1$. For the proof we make use of the following building blocks:

- Let $\mathsf{ECC}\colon \{0,1\}^{d_2} \to \colon \{0,1\}^{D_2}$ be the error correcting code from Theorem 4.4 set with relative distance $1/4$. By Theorem 4.4, $D_2 = O(d_2)$.
- Let $r = \log_{4/3}(1/\varepsilon)$ and set $v = r \cdot \log_2 D_2$. Let $\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^{d_1} \to \{0,1\}^v$ be the extractor from Theorem 4.2, set with error ε. Note that we set d_1 to be large enough as required by a seed for Ext. Moreover, one can verify that by our assumption on k, given by Equation (10), $k \geq 2v$ as required by Theorem 4.2.

We proceed by proving the following claim.

▶ **Claim 10.5.** *The function*

$$\mathsf{AdvGen}(x,y) = \mathsf{NMExt}(x,y_1) \circ \mathsf{ECC}(y_2)_{\mathsf{Ext}(x,y_1)}$$

is a d_1-nice t-advice generator for entropy k with error $O(t\sqrt{\varepsilon})$.

Proof. Let X be an (n,k)-source, let Y be a random variable that is independent of X and is uniformly distributed over $\{0,1\}^d$, and let $\{A_i\colon \{0,1\}^d \to \{0,1\}^d\}_{i=1}^t$ be functions with no fixed points. By Lemma 4.14, there exists a set $B \subset \{0,1\}^{d_1}$ of density $\sqrt{\varepsilon}$ such that for any $y_1 \notin B$ and any d_1-bit string $y_1' \neq y_1$, it holds that

$$(\mathsf{NMExt}(X,y_1), \mathsf{NMExt}(X,y_1')) \approx_{\sqrt{\varepsilon}} (U,\cdot)\,. \tag{13}$$

We now fix $y_1 \sim Y_1$ and $y_1^i \sim (\mathcal{A}_i(Y))_1$ for $i = 1,\ldots,t$. Clearly, these fixings do not introduce dependencies between X,Y. Furthermore, by the above, we can aggregate $\sqrt{\varepsilon}$ to the total error and assume that $y_1 \notin B$. Let I be the set of $i \in [t]$ such that $y_1 \neq y_1^i$.

Fix $i \in I$. By Equation (13) it holds that $\mathsf{NMExt}(X,y_1)$ is $\sqrt{\varepsilon}$-close to uniform even conditioned on $\mathsf{NMExt}(X,y_1^i)$. In particular, as NMExt has output length $\log(1/\varepsilon)$, the probability that $\mathsf{NMExt}(X,y_1) = \mathsf{NMExt}(X,y_1^i)$ is bounded above by $O(\sqrt{\varepsilon})$. By the union bound over all $i \in I$, we have that except with probability $O(t\sqrt{\varepsilon})$, for all $i \in I$, $\mathsf{AdvGen}(X,Y) \neq \mathsf{AdvGen}(X,\mathcal{A}_i(Y))$.

Consider now $i \notin I$. Conditioned on $y_1 = y_1^i$ we have that $Y_2 \neq (\mathcal{A}_i(Y))_2$, and so the codewords $\mathsf{ECC}(Y_2), \mathsf{ECC}((\mathcal{A}_i(Y))_2)$ agree on at most $3/4$ of the coordinates of $[D_2]$. Hence, the set of r-tuples over $[D_2]$ for which $\mathsf{ECC}(Y_2)$ agrees with $\mathsf{ECC}((\mathcal{A}_i(Y))_2)$ on all r coordinates of the tuple has density at most $(3/4)^r = \varepsilon$ within $[D_2]^r$. By the union bound over all $i \notin I$, at most εt fraction of the r-tuples in $[D_2]^r$ are such that $\mathsf{ECC}(Y_2)$ agrees with $\mathsf{ECC}((\mathcal{A}_i(Y))_2)$ for some $i \notin I$. We denote this set of r-tuples by $B' \subseteq [D_2]^r$.

Recall that Ext is a strong seeded extractor with error ε, and so except for probability $\sqrt{\varepsilon}$ over the choices of y_1, we have that $\mathsf{Ext}(X,y_1)$ is $\sqrt{\varepsilon}$-close to uniform. For any such y_1 we have that $\mathbf{Pr}[\mathsf{Ext}(X,y_1) \in B'] \leq \varepsilon t + O(\sqrt{\varepsilon})$. Thus, except with probability $O(t\sqrt{\varepsilon})$ we have that also for all $i \notin I$, $\mathsf{AdvGen}(X,Y) \neq \mathsf{AdvGen}(X,\mathcal{A}_i(Y))$.

As for niceness property, by Lemma 4.8, the fixings of $Y_1, \{(\mathcal{A}_i(Y))_1\}_{i=1}^t$ reduce the average min-entropy of Y_2 by a most $(t+1)d_1$. Once $Y_1, \{(\mathcal{A}_i(Y))_1\}_{i=1}^t$ are fixed, we have that $\mathsf{Ext}(X,Y_1)$, $\{\mathsf{Ext}(X,(\mathcal{A}_i(Y))_1)\}_{i=1}^t$, $\mathsf{NMExt}(X,Y_1)$, and $\{\mathsf{NMExt}(X,(\mathcal{A}_i(Y))_1)\}_{i=1}^t$ are all deterministic functions of X. Thus, we can fix the latter random variables without introducing dependencies between X,Y. Further, by Lemma 4.8, the average min-entropy of X decreases by at most $O(t \cdot \log(1/\varepsilon) \cdot \log d)$.

After these fixings, $\mathsf{ECC}(Y_2)_{\mathsf{Ext}(X,Y_1)}$ and $\{\mathsf{ECC}((\mathcal{A}_i(Y))_2)_{\mathsf{Ext}(X,(\mathcal{A}_i(Y))_1)}\}_{i=1}^t$ are deterministic functions of Y, each consists of r bits. Thus, fixing these random variables will reduce the average min-entropy of Y by at most $(t+1)r = O(t \cdot \log(1/\varepsilon))$. Further, these fixings do not introduce any dependencies between X, Y. Note that after all of the fixings done so far, $\mathsf{AdvGen}(X,Y)$ and $\{\mathsf{AdvGen}(X,(\mathcal{A}_i(Y)))\}_{i=1}^t$ are all fixed.

To summarize, in the process of fixing $\mathsf{AdvGen}(X,Y)$, $\{\mathsf{AdvGen}(X,(\mathcal{A}_i(Y)))\}_{i=1}^t$, the random variable Y_2 lost an average entropy of $(t+1)(d_1+r)$. As we set $d_2 = 500t^2d_1$ we have that except with probability ε over these fixings, Y_2 has min-entropy rate $1 - 1/(100t)$. As for X, the fixings reduced its average min-entropy by $O(t \cdot \log(1/\varepsilon) \cdot \log d)$, and so except with probability ε, X has min-entropy rate 0.99 conditioned on these fixings. ◀

To conclude the proof we apply Lemma 10.4 with AdvGen defined above and the parameter m. The hypothesis of Lemma 10.4 is met due to our hypothesis on m, d, k and since $a = O(\log(1/\varepsilon))$. ◀

Acknowledgements. We wish to thank Ronen Shaltiel for enjoyable conversations we had regarding this work during his visit at Caltech.

References

1. D. Aggarwal, K. Hosseini, and S. Lovett. Affine-malleable extractors, spectrum doubling, and application to privacy amplification. In *Electronic Colloquium on Computational Complexity (ECCC)*, page 179, 2015.
2. B. Barak, G. Kindler, R. Shaltiel, B. Sudakov, and A. Wigderson. Simulating independence: New constructions of condensers, Ramsey graphs, dispersers, and extractors. In *Proceedings of the thirty-seventh annual ACM Symposium on Theory of Computing*, pages 1–10. ACM, 2005.
3. B. Barak, A. Rao, R. Shaltiel, and A. Wigderson. 2-source dispersers for $n^{o(1)}$ entropy, and Ramsey graphs beating the Frankl-Wilson construction. *Annals of Mathematics*, 176(3):1483–1544, 2012.
4. E. Chattopadhyay, V. Goyal, and X. Li. Non-malleable extractors and codes, with their many tampered extensions. *arXiv preprint arXiv:1505.00107*, 2015.
5. E. Chattopadhyay and D. Zuckerman. Explicit two-source extractors and resilient functions. *Electronic Colloquium on Computational Complexity (ECCC)*, 2015.
6. B. Chor and O. Goldreich. Unbiased bits from sources of weak randomness and probabilistic communication complexity. *SIAM Journal on Computing*, 17(2):230–261, 1988.
7. G. Cohen. Local correlation breakers and applications to three-source extractors and mergers. In *Electronic Colloquium on Computational Complexity (ECCC)*, page 38, 2015.
8. G. Cohen. Two-source dispersers for polylogarithmic entropy and improved Ramsey graphs. *arXiv preprint arXiv:1506.04428*, 2015.
9. G. Cohen, R. Raz, and G. Segev. Nonmalleable extractors with short seeds and applications to privacy amplification. *SIAM Journal on Computing*, 43(2):450–476, 2014.
10. Y. Dodis, X. Li, T. D. Wooley, and D. Zuckerman. Privacy amplification and nonmalleable extractors via character sums. *SIAM Journal on Computing*, 43(2):800–830, 2014.
11. Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM Journal on Computing*, 38(1):97–139, 2008.
12. Y. Dodis and D. Wichs. Non-malleable extractors and symmetric key cryptography from weak secrets. In *Proceedings of the forty-first annual ACM Symposium on Theory of Computing*, pages 601–610. ACM, 2009.

13 Z. Dvir, S. Kopparty, S. Saraf, and M. Sudan. Extensions to the method of multiplicities, with applications to Kakeya sets and mergers. In *50th Annual IEEE Symposium on Foundations of Computer Science*, pages 181–190. IEEE, 2009.

14 S. Dziembowski and K. Pietrzak. Intrusion-resilient secret sharing. In *48th Annual IEEE Symposium on Foundations of Computer Science*, pages 227–237, 2007.

15 V. Guruswami, C. Umans, and S. Vadhan. Unbalanced expanders and randomness extractors from Parvaresh–Vardy codes. *Journal of the ACM*, 56(4):20, 2009.

16 X. Li. Improved constructions of three source extractors. In *IEEE 26th Annual Conference on Computational Complexity*, pages 126–136, 2011.

17 X. Li. Design extractors, non-malleable condensers and privacy amplification. In *Proceedings of the forty-fourth annual ACM Symposium on Theory of Computing*, pages 837–854, 2012.

18 X. Li. Non-malleable condensers for arbitrary min-entropy, and almost optimal protocols for privacy amplification. *arXiv preprint arXiv:1211.0651*, 2012.

19 X. Li. Non-malleable extractors, two-source extractors and privacy amplification. In *IEEE 53rd Annual Symposium on Foundations of Computer Science*, pages 688–697, 2012.

20 X. Li. Extractors for a constant number of independent sources with polylogarithmic min-entropy. In *IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 100–109, 2013.

21 X. Li. New independent source extractors with exponential improvement. In *Proceedings of the forty-fifth annual ACM Symposium on Theory of Computing*, pages 783–792. ACM, 2013.

22 X. Li. Extractors for affine sources with polylogarithmic entropy. In *Electronic Colloquium on Computational Complexity (ECCC)*, page 121, 2015.

23 X. Li. Improved constructions of two-source extractors. In *Electronic Colloquium on Computational Complexity (ECCC)*, page 125, 2015.

24 X. Li. Three-source extractors for polylogarithmic min-entropy. *Electronic Colloquium on Computational Complexity (ECCC)*, 2015.

25 X. Li, T. D. Wooley, and D. Zuckerman. Privacy amplification and nonmalleable extractors via character sums. *arXiv preprint arXiv:1102.5415*, 2011.

26 N. Nisan and D. Zuckerman. Randomness is linear in space. *Journal of Computer and System Sciences*, 52(1):43–52, 1996.

27 A. Rao. Extractors for a constant number of polynomially small min-entropy independent sources. *SIAM Journal on Computing*, 39(1):168–194, 2009.

28 R. Raz. Extractors with weak random seeds. In *Proceedings of the thirty-seventh annual ACM Symposium on Theory of Computing*, pages 11–20, 2005.

29 O. Reingold, R. Shaltiel, and A. Wigderson. Extracting randomness via repeated condensing. In *Proceedings. 41st Annual Symposium on Foundations of Computer Science, 2000*, pages 22–31. IEEE, 2000.

30 R. Shaltiel. An introduction to randomness extractors. In *Automata, languages and programming*, pages 21–41. Springer, 2011.

31 A. Ta-Shma and C. Umans. Better condensers and new extractors from Parvaresh-Vardy codes. In *IEEE 27th Annual Conference onComputational Complexity (CCC)*, pages 309–315. IEEE, 2012.

32 Salil P. Vadhan. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 2011.

33 D. Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. *Theory of Computing*, 3:103–128, 2007.

Pseudorandomness When the Odds are Against You

Sergei Artemenko[*1], Russell Impagliazzo[†2], Valentine Kabanets[‡3], and Ronen Shaltiel[§4]

1 **Department of Computer Science, University of Haifa, Haifa, Israel**
sartemen@gmail.com
2 **Department of Computer Science, University of California, San Diego, USA**
russell@cs.ucsd.edu
3 **School of Computing Science, Simon Fraser University, Burnaby, Canada**
kabanets@cs.sfu.ca
4 **Department of Computer Science, University of Haifa, Haifa, Israel**
ronen@cs.haifa.ac.il

Abstract

Impagliazzo and Wigderson [25] showed that if E = DTIME($2^{O(n)}$) requires size $2^{\Omega(n)}$ circuits, then every time T constant-error randomized algorithm can be simulated deterministically in time poly(T). However, such polynomial slowdown is a deal breaker when $T = 2^{\alpha \cdot n}$, for a constant $\alpha > 0$, as is the case for some randomized algorithms for NP-complete problems. Paturi and Pudlak [30] observed that many such algorithms are obtained from randomized time T algorithms, for $T \leq 2^{o(n)}$, with large one-sided error $1-\epsilon$, for $\epsilon = 2^{-\alpha \cdot n}$, that are repeated $1/\epsilon$ times to yield a constant-error randomized algorithm running in time $T/\epsilon = 2^{(\alpha + o(1)) \cdot n}$.

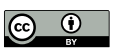
We show that if E requires size $2^{\Omega(n)}$ nondeterministic circuits, then there is a poly(n)-time ϵ-HSG (Hitting-Set Generator) $H\colon \{0,1\}^{O(\log n)+\log(1/\epsilon)} \to \{0,1\}^n$, implying that time T randomized algorithms with one-sided error $1-\epsilon$ can be simulated in deterministic time poly(T)/ϵ. In particular, under this hardness assumption, the fastest known constant-error randomized algorithm for k-SAT (for $k \geq 4$) by Paturi et al. [31] can be made deterministic with essentially the same time bound. This is the first hardness versus randomness tradeoff for algorithms for NP-complete problems. We address the necessity of our assumption by showing that HSGs with very low error imply hardness for nondeterministic circuits with "few" nondeterministic bits.

Applebaum et al. [2] showed that "black-box techniques" cannot achieve poly(n)-time computable ϵ-PRGs (Pseudo-Random Generators) for $\epsilon = n^{-\omega(1)}$, even if we assume hardness against circuits with oracle access to an arbitrary language in the polynomial time hierarchy. We introduce weaker variants of PRGs with *relative error*, that do follow under the latter hardness assumption. Specifically, we say that a function $G : \{0,1\}^r \to \{0,1\}^n$ is an (ϵ, δ)-re-PRG for a circuit C if $(1-\epsilon) \cdot \Pr[C(U_n) = 1] - \delta \leq \Pr[C(G(U_r) = 1] \leq (1+\epsilon) \cdot \Pr[C(U_n) = 1] + \delta$. We construct poly($n$)-time computable (ϵ, δ)-re-PRGs with arbitrary polynomial stretch, $\epsilon = n^{-O(1)}$ and $\delta = 2^{-n^{\Omega(1)}}$. We also construct PRGs with relative error that fool *non-boolean distinguishers* (in the sense introduced by Dubrov and Ishai [11]).

Our techniques use ideas from [30, 43, 2]. Common themes in our proofs are "composing" a PRG/HSG with a combinatorial object such as dispersers and extractors, and the use of nondeterministic reductions in the spirit of Feige and Lund [12].

1998 ACM Subject Classification F.1.2 Modes of Computation

[*] Research supported by ERC starting grant 279559.
[†] Research supported by the Simons Foundation and NSF grants #CNS-1523467 and CCF-121351.
[‡] Research supported by an NSERC Discovery grant.
[§] Research supported by BSF grant 2010120, ISF grant 864/11, and ERC starting grant 279559.



COMPUTATIONAL COMPLEXITY CONFERENCE

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 9; pp. 9:1–9:35

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

Keywords and phrases Derandomization, pseudorandom generator, hitting-set generator, relative error

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.9

1 Introduction

Derandomization, the construction of deterministic algorithms from randomized algorithms, is an area where there are tight connections between lower bounds and algorithm design. Indeed, strong enough circuit lower bounds can be used to construct pseudo-random generators that can then be used to simulate randomized algorithms with only polynomial overhead. This is often summarized as saying, "Randomness is never essential for efficient algorithm design", if such lower bounds exists.

However, there are many algorithmic applications where a simulation with polynomial overhead is next to useless. For example, consider the best algorithms for different NP-complete problems such as different variations of SAT. Many of the best algorithms for these problems are in fact randomized or careful derandomizations of probabilistic algorithms [32, 31, 34, 22, 33, 1]. If the Exponential Time Hypothesis is true, these problems all require exponential time, so a polynomial slowdown might take an algorithm from best possible to worse than exhaustive search. On the other hand, as observed in [30], most of these randomized algorithms are in fact fast algorithms, but with only a very small success probability ϵ. [30] call such algorithms OPP algorithms, for *One-sided error Probabilistic Polynomial Time*. (These algorithms can then be repeated $O(1/\epsilon)$ times to yield a final randomized algorithm with constant success probability).

It is not to hard to see that OPP algorithms can be derandomized in time comparable to the running time of the final randomized algorithm, if we can construct efficient pseudorandom generators (or hitting set generators) which work for a very low error parameter ϵ using short seeds.

In this paper, we address the question of constructing such generators. We give constructions of pseudorandom generators and hitting-set generators that get essentially optimal simulations of OPP, and go beyond to also consider algorithms that have two-sided error that only slightly favors the correct answer. In order to get these generators, we need stronger lower bounds, lower bounds against nondeterministic circuits rather than deterministic circuits. However, we also show that such lower bounds are necessary for strong derandomization of OPP algorithms.

As we explain later, in some settings there are black-box impossibility results on constructing generators for very low error parameter. In this paper, we also introduce new notions of pseudorandom generator with "relative error" which can be used to replace low-error generators in certain settings. We give constructions of such generators, and discuss potential applications.

1.1 Pseudorandom generators and hitting-set generators

We start by reviewing the definitions of pseudorandom generators and hitting set generators.

▶ **Definition 1.1** (PRGs and HSGs)**.** Let $\mathcal{C}$ be a class of boolean functions $C : \{0,1\}^n \to \{0,1\}$. A function $G : \{0,1\}^r \to \{0,1\}^n$ is:

- an ϵ-PRG for $\mathcal{C}$ if for every C in $\mathcal{C}$, $|\Pr[C(G(U_r)) = 1] - \Pr[C(U_n) = 1]| \leq \epsilon$.
- an ϵ-HSG for $\mathcal{C}$ if for every C in $\mathcal{C}$ s.t. $\Pr[C(U_n) = 1] > \epsilon$, there exists $x \in \{0,1\}^r$ s.t. $C(G(x)) = 1$.

We will be interested in generators that fool circuits of size n^b for some fixed constant b, and run in time poly(n^b). In the case of logarithmic seed length ($r = O(\log n)$) this is often referred to as the Nisan-Wigderson setting. We will typically be interested in larger seed length (which is required to handle low error ϵ).

Such PRGs imply circuit lower bounds, and so in the current state of knowledge, we cannot construct them unconditionally. A long line of research [9, 45, 29, 5, 21, 25, 41, 24, 35, 44] is devoted to constructing such PRGs under the weakest possible hardness assumptions. An important milestone of this line of research is the hardness versus randomness tradeoff of Impagliazzo and Wigderson [25].

▶ **Definition 1.2** (E is hard for exponential size circuits)**.** We say that E is hard for exponential size circuits, if there exists a language L in $\mathrm{E} = \mathrm{DTIME}(2^{O(n)})$ 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.

▶ **Theorem 1.3** ([25])**.** *If E is hard for exponential size circuits, then for every constant $b > 1$ there exists a constant $c > 1$ such that for every sufficiently large n, there is a function $G : \{0,1\}^r \to \{0,1\}^n$ that is an n^{-b}-PRG for size n^b circuits, with $r = c \log n$. Furthermore, G is computable in time poly(n^b).*

By a standard probabilistic argument, for every $\epsilon > 0$, there exists a (nonexplicit) ϵ-PRG with seed length $r = c \log n + O(\log(1/\epsilon))$ for size n^b circuits. In particular, if we shoot for PRGs with "polynomial stretch" (that is, $r = n^{\Omega(1)}$), we can expect to get error $\epsilon = 2^{-n^{\Omega(1)}}$ that is exponentially small. The known proofs of Theorem 1.3 do not achieve these parameters. In fact, they cannot achieve negligible error of $\epsilon = n^{-\omega(1)}$ if the constructed PRG runs in time poly(n), even if we allow large seed length $r = \Omega(n)$.[1] It is natural to ask if we can construct poly(n)-time computable ϵ-PRGs or ϵ-HSGs with $\epsilon = n^{-\omega(1)}$? Under what assumptions? Can we get ϵ to be exponentially small?

1.2 Limitations on deterministic reductions for PRGs and HSGs

There is a formal sense in which "black-box" proofs of Theorem 1.3 cannot achieve negligible ϵ [38, 18, 3]. It is instructive to explain this argument. Loosely speaking, "black-box" proofs are made of two components: The first is a construction, this is an oracle procedure $\mathrm{Con}^{(\cdot)}$ which implements the PRG $G(x) = \mathrm{Con}^f(x)$ given oracle access to the hard function f that is guaranteed in the hardness assumption. Note that as G runs in time poly(n), the construction cannot afford to query $f \in \mathrm{E}$ on inputs of length larger than $\ell = c \log n$ (for some constant $c > 1$). On inputs of this length, the maximal possible circuit complexity of f is at most $2^\ell = n^c$.

[1] In this paper we are mostly interested in PRGs that run in time poly(n), that is polynomial in the output length. Another natural notion is allowing PRGs to run in time exponential in the seed length (that is time $2^{O(r)}$). These notions differ in the case of "polynomial stretch" ($r = n^{\Omega(1)}$) which will be the setting that we will consider. PRGs which run in time exponential in the seed length make sense in most applications which run the PRGs over all 2^r seeds. An exception is the case of "SAT algorithms" where r may be $\alpha \cdot n$ for a constant $\alpha < 1$ that is close to 1, and there may be a substantial difference between running 2^r instantiations of a time poly(n) PRG, (which gives time less than 2^n) compared to 2^r instantiations of a PRG running in time larger than 2^r, which takes time at least $2^r \cdot 2^r > 2^n$. Other applications in which there is a big difference between $2^{O(r)}$ time PRGs and poly(n) time PRGs are applications that run the PRG only once. In such cases, it is typically important that the PRG run in time polynomial in the output length (so that the application runs in polynomial time). We will elaborate on several such applications in this paper.

The second component in the proof is a reduction $\text{Red}^{(\cdot)}$, which is given black-box access to a circuit D that is not fooled by the PRG, and implements a small circuit $C : \{0,1\}^\ell \to \{0,1\}$ for f (contradicting the hardness assumption). By the discussion above, in order to contradict the hardness assumption, the reduction must produce a circuit C of size less than n^c. However, note that in some sense, the reduction needs to distinguish between a useless function D that always answers zero, and a useful function D that is not fooled by the PRG, and answers one with probability ϵ. It intuitively follows that Red (which only has black-box access to D) must query D at least $1/\epsilon$ times. (This part of the argument can be made formal, see [38, 18, 3], and also applies for constructions of HSGs). In particular, the circuit C that is implemented by Red has size $\geq 1/\epsilon$. This gives $1/\epsilon \leq n^c$ which implies $\epsilon \geq n^{-c}$.

The same kind of limitations apply in the closely related problem of hardness amplification (there the goal is to start with a worst-case hard lower bound (such as the assumption E is hard for exponential size circuits) and produce an average-case hard function. An influential work of Feige and Lund [12] shows that nondeterministic reductions can be used to bypass these limitations. Specifically, we may relax the requirement that Red implements a (deterministic) circuit, and allow Red to implement a nondeterministic circuit. Indeed, nondeterminism allows Red to make exponentially many queries to D (on different "nondeterministic computations paths") circumventing the limitation above. The price we pay is that we need to assume a hardness assumption against nondeterministic circuits. This approach indeed leads to hardness amplification with negligible ϵ under hardness assumptions for nondeterministic circuits [43, 10, 2].

1.3 Hardness assumptions for nondeterministic circuits

We start by defining various notions of nondeterministic circuit.

▶ **Definition 1.4** (nondeterministic circuits with few nondeterministic bits)**.** We say that a function $f : \{0,1\}^n \to \{0,1\}$ is computed by a size s circuit D with k nondeterministic bits if there exists a size s deterministic circuit $C : \{0,1\}^n \times \{0,1\}^k \to \{0,1\}$ such that for every $x \in \{0,1\}^n$

$$f(x) = 1 \;\Leftrightarrow \exists y \in \{0,1\}^k \text{ s.t. } D(x,y) = 1.$$

▶ **Definition 1.5** (oracle circuits and Σ_i-circuits)**.** 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 Σ_i-circuit is an A-circuit where A is the canonical Σ_i^P-complete language. The size of all circuits is the total number of wires and gates.[2]

Note for example that an NP-circuit is different than a nondeterministic circuit. The former is a nonuniform analogue of P^{NP} (which contains coNP) while the latter is an analogue of NP. Hardness assumptions against nondeterministic/NP/Σ_i circuits appear in the literature in various contexts of derandomization [27, 28, 43, 15, 35, 19, 36, 6, 37, 10, 4, 2]. Typically, the assumption is of the following form: E is hard for exponential size circuits (where the type of circuits is one of the types discussed above). More specifically:

[2] 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 $s^{\Theta(1)}$ Σ_i-circuit is equivalent to $\text{DTIME}^{\Sigma_i^P}(s^{\Theta(1)})/s^{\Theta(1)}$.

▶ **Definition 1.6.** We say that E is hard for exponential size circuits of type X if there exists a problem L in $\mathrm{E} = \mathrm{DTIME}(2^{O(n)})$ 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 EXP $\neq$ NP or EXP $\neq \Sigma_2^{\mathrm{P}}$ (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.[3]

It is known that Theorem 1.3 extends to every type of circuits considered in Definitions 1.4, Definition 1.5 and their combinations.

▶ **Theorem 1.7** ([25, 27, 35, 36])**.** *For every $i \geq 0$, the statement of Theorem 1.3 also holds if we replace every occurrence of the word "circuits" by "Σ_i-circuits" or alternatively by "nondeterministic Σ_i-circuits".*

1.4 A construction of HSGs with low error

Our first result is a construction of a $\mathrm{poly}(n)$-time computable ϵ-HSG that works for small ϵ. We rely on the assumption that E is hard for exponential size nondeterministic circuits. (Note that by the earlier discussion we cannot expect to get this with hardness against deterministic circuits).

▶ **Theorem 1.8** (HSG with seed length $r = \log(1/\epsilon) + O(\log n)$)**.** *If E is hard for exponential size nondeterministic circuits then for every constant $b > 1$ there exists a constant $c > 1$ such that for every sufficiently large n, there is a function $G : \{0,1\}^r \to \{0,1\}^n$ that is an ϵ-HSG for size n^b circuits, with $r = \log(1/\epsilon) + c \log n$. Furthermore, G is computable in time poly(n^b).*

We stress that the seed length achieved in Theorem 1.8 matches that of nonexplicit HSGs that exist by a probabilistic argument: the dependence of r on ϵ is an additive factor of $1 \cdot \log(1/\epsilon)$. In some settings, achieving this correct dependence (with the right constant) for a polynomial time computable HSG is crucial (as seen in the example of the next section).[4]

1.5 Derandomizing randomized algorithms with large one sided error

By going over all seeds of the HSG, we can deterministically simulate randomized polynomial time algorithms with large one-sided error of $1 - \epsilon(n)$ in time $2^r \cdot \mathrm{poly}(n) = \mathrm{poly}(n)/\epsilon(n)$. This is stated precisely in the theorem below.

▶ **Theorem 1.9.** *Let A be a time $T(n) \geq n$ randomized algorithm that accepts some language L with one sided error of $1 - \epsilon(n)$. That is, for every sufficiently large n and $x \in \{0,1\}^n$:*

- $x \in L \Rightarrow \Pr[A(x) = 1] \geq \epsilon(n)$.

[3] 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.

[4] We remark that when applying the probabilistic argument for PRGs we get an additive factor of $2 \cdot \log(1/\epsilon)$ whereas for HSGs it is possible to get $1 \cdot \log(1/\epsilon)$. This difference is crucial for the application of derandomizing OPP algorithms as we explain below.

- $x \notin L \Rightarrow \Pr[A(x) = 1] = 0$.

If E is hard for exponential size nondeterministic circuits then there is a deterministic algorithm running in time poly$(T(n))/\epsilon(n)$ that accepts L.

Note that if $T(n) \ll 1/\epsilon(n)$ then the slowdown is polynomial in $T(n)$ but *linear* in $1/\epsilon(n)$. As we explain below, in many algorithms in the literature, $T(n) = \text{poly}(n)$ and $\epsilon(n) = 2^{-\alpha \cdot n}$ for some constant $0 < \alpha < 1$. Note that even the more modest goal of amplifying the success probability of A to obtain a randomized algorithm with constant one-sided error, requires running time of $\text{poly}(T(n))/\epsilon(n)$ which is $\text{poly}(n) \cdot 2^{\alpha \cdot n}$ for the choices above. We achieve the same time with a deterministic algorithm.[5]

Moreover, if we were to amplify A, and then derandomize it using known hardness versus randomness tradeoffs, we would end up with a deterministic algorithm running in time at least $\text{poly}(n)/\epsilon(n)^c$ for a large constant c. Such a slowdown is a "deal breaker" if ϵ is very small (say $\epsilon = 2^{-\alpha \cdot n}$) for a constant α that is only slightly smaller than one. In the next section we observe that this is the case in many randomized k-SAT algorithms.

1.6 Deterministic k-SAT algorithms

Paturi and Pudlak [30] observed that many of the randomized algorithms in the literature for solving k-SAT and other NP-complete problems (in particular the algorithm of Paturi, Pudlak and Zane [32], Paturi et al. [31], Schöning [34]) are based on designing probabilistic polynomial-time (or subexponential-time) algorithms with one-sided error, whose success probability may be exponentially small. To improve the success probability to a constant, one repeats the original randomized algorithm the inverse of the success probability times. The running time of this new randomized algorithm is dominated by the inverse of the success probability of the original algorithm.

For example, suppose A is a SAT-algorithm running in time $T(n) = 2^{o(n)}$ that, given a satisfiable formula, produces a satisfying assignment with probability at least $\epsilon = 2^{-\alpha \cdot n}$ (for some constant $0 < \alpha < 1$). The algorithm with constant success probability is produced by repeating A $O(1/\epsilon)$ times, and so has the running time $2^{(\alpha + o(1)) \cdot n}$.

By Theorem 1.9, all such algorithms can be made deterministic (with essentially the same time bounds) under the assumption that E is hard for nondeterministic circuits. This is the first application of the hardness versus randomness paradigm that yields a nontrivial derandomization of these algorithms.

Some of these randomized algorithms (and in particular the PPZ algorithm [32] and Schöning's algorithm [34]) have deterministic versions. However the fastest known algorithms for k-SAT for $k \geq 4$ due to Paturi et al. [31] does not have a matching deterministic algorithm. We get the first derandomization result for these k-SAT algorithms from [31], based on circuit complexity assumptions.

For each $k \geq 4$, let us denote by $T_k^{\text{PPSZ}}(n) \leq 2^{o(n)}$ the running time of the randomized PPSZ algorithm [31], and let $2^{-\alpha_k^{\text{PPSZ}} \cdot n}$ (where $0 < \alpha_k^{\text{PPSZ}} < 1$ is a constant specified in [31]) be its success probability. The fastest known constant-error randomized algorithm for k-SAT, for $k \geq 4$, is obtained by repeating the above algorithm the inverse success probability number of times, resulting in the running time

$$2^{\alpha_k^{\text{PPSZ}} \cdot n} \cdot T_k^{\text{PPSZ}}(n) \leq 2^{\left(\alpha_k^{\text{PPSZ}} + o(1)\right) \cdot n}.$$

[5] The assumption in Theorem 1.9 can be relaxed to "E is hard for size $n^{\omega(1)}$ nondeterministic circuits" and then, the final running time will be $2^{T(n)^{o(1)}}/\epsilon(n)$.

Our approach gives the following result:

▶ **Theorem 1.10.** *If E is hard for nondeterministic circuits, then there are deterministic algorithms for k-SAT, for each $k \geq 4$, running in time*

$$T_k(n) = 2^{\alpha_k^{PPSZ} \cdot n} \cdot poly(T_k^{PPSZ}(n)) \leq 2^{\left(\alpha_k^{PPSZ} + o(1)\right) \cdot n}.$$

We remark that the assumption could be relaxed to E is hard for size $n^{\omega(1)}$ nondeterministic circuits, and then the deterministic time $T_k(n)$ for k-SAT would become

$$2^{\alpha_k^{\mathrm{PPSZ}} \cdot n} \cdot 2^{\left(T_k^{\mathrm{PPSZ}}(n)\right)^{o(1)}},$$

which is still at most $2^{\left(\alpha_k^{\mathrm{PPSZ}} + o(1)\right) \cdot n}$, as $T_k^{\mathrm{PPSZ}}(n) \leq n^{\beta(n)}$ for every $\beta(n) \in \omega(1)$.

1.7 Hardness assumptions implied by HSGs with low error

In Theorem 1.8 we show that hardness for nondeterministic circuits implies HSGs with low error. Is this assumption necessary? Is the converse statement true? We do not know the answer to these questions. However, we can show ϵ-HSGs for deterministic poly-size circuits are essentially equivalent to $\frac{1}{2}$-HSGs for a subclass of nondeterministic circuits: The class of poly-size nondeterministic circuits with approximately $\log(1/\epsilon)$ nondeterministic bits. The precise definitions and statements appear in Section 6. Note that as $n > r \geq \log(1/\epsilon)$, the circuits we are interested in are nondeterministic circuits with a sublinear number of nondeterministic bits. Using this connection, we can show that ϵ-HSGs with seed length $r = n^{o(1)} + O(\log(1/\epsilon))$ imply that E is hard for poly-size nondeterministic circuits with $o(n)$ nondeterministic bits.

▶ **Theorem 1.11.** *Let $\delta > 0$ be a constant. Assume that for every sufficiently large n, there is a 2^{-n^δ}-HSG $H : \{0,1\}^{O(n^\delta)} \to \{0,1\}^n$ for size $s \geq n$ circuits, and furthermore that the family of functions $H = \{H_n\}$ is computable in time exponential in the seed length, that is time $2^{O(n^\delta)}$. Then there exists a constant $\gamma > 0$ and a problem $L \in E$ such that, for every sufficiently large n', nondeterministic circuits of size $(\gamma n')^{1/\delta}$ with $\gamma \cdot n'$ nondeterministic bits fail to compute the characteristic function of L on inputs of length n'.*

1.8 Limitations on nondeterministic reductions for PRGs

Theorem 1.8 demonstrates that hardness assumption for nondeterministic circuits can yield polynomial time computable HSGs with low error. A recent result of Applebaum et al. [2] shows that these techniques cannot be extended to yield PRGs. We state this result informally below (the reader is referred to [2] for the formal model and precise statement).

▶ **Informal Theorem 1.12.** *For every $i \geq 0$, it is impossible to use "black-box reductions" to prove that the assumption that E is hard for exponential size Σ_i-circuits implies that for $\epsilon = n^{-\omega(1)}$, there is a poly$(n)$-time computable ϵ-PRG $G : \{0,1\}^{n-1} \to \{0,1\}^n$ for size n^2.*

While hardness against circuits with oracle to PH problems does not suffice, hardness for circuits with oracle to PSPACE problems does suffice. This follows by inspecting the correctness proofs of Theorem 1.3 (the one that seems easiest to handle is by Sudan, Trevisan and Vadhan [41]).

▶ **Theorem 1.13** (PRG with seed length $r = O(\log n) + \log(1/\epsilon)$)**.** *If E is hard for exponential size PSPACE-circuits then for every constant $b > 1$ there exists a constant $c > 1$ such that for every sufficiently large n, there is a function $G : \{0,1\}^r \to \{0,1\}^n$ that is an ϵ-PRG for size n^b circuits, with $c \cdot (\log n + \log(1/\epsilon))$. Furthermore, G is computable in time poly(n^b).*

In fact, something more precise can be said. The circuit model that comes up is the nonuniform class which corresponds to the fourth level of the counting hierarchy (which is contained in PSPACE).

1.9 Derandomizing randomized algorithms with large two sided error

By Theorem 1.12 we do not expect to construct ϵ-PRGs for small ϵ under the assumption that E is hard for Σ_1-circuits. Nevertheless, it turns out that we can use this assumption to extend Theorem 1.9 to the case of two-sided error.

▶ **Theorem 1.14.** *Let A be a time $T(n) \geq n$ randomized algorithm such that for every sufficiently large n and $x \in \{0,1\}^n$:*

- $x \in L \Rightarrow \Pr[A(x) = 1] \geq 2 \cdot \epsilon(n)$.
- $x \notin L \Rightarrow \Pr[A(x) = 1] \leq \epsilon(n)$.

If E is hard for exponential size Σ_1-circuits then there is a deterministic algorithm running in time $\frac{poly(T(n))}{\epsilon(n)^2}$ that accepts L.[6]

Note that even the more modest goal of amplifying the success probability of A to obtain a randomized algorithm with constant two-sided error, requires running time of $T(n)/\epsilon(n)^2$. We achieve roughly the same time with a deterministic algorithm. In fact, the conclusion of Theorem 1.14 is stronger than the one that follows if we were to run the PRG of Theorem 1.13 on all seeds. The latter approach would have given time $\frac{\text{poly}(n)}{\epsilon(n)^c}$ for a large constant c.

Loosely speaking, we avoid the limitations on PRGs by showing a derandomization procedure which runs the algorithm on 2^r "pseudorandom strings" (just like in PRGs). The key difference is that the "estimation of the success probability of $A(x)$" is not done by "averaging over all pseudorandom strings". This allows the procedure not to be fooled by a small fraction of "pseudorandom strings" that yield incorrect results.

1.10 Implications to derandomization of BPP_{path}

The class BPP_{path} defined by Han, Hemaspaandra and Thierauf [20] consists of polynomial time randomized algorithms $A(x)$ which are allowed to output "don't know". It is required that for every input x, conditioned on giving an answer, the probability that $A(x)$ answers correctly is at least $2/3$, and that the probability that $A(x)$ answers is larger than $\epsilon(n)$ for some $\epsilon(n) > 0$.

Han, Hemaspaandra and Thierauf [20] showed that this class is quite powerful and contains $\text{P}_{||}^{\text{NP}}$ which contains NP. (The subscript "$||$" in $\text{P}_{||}^{\text{NP}}$ means that the queries to the NP oracle are nonadaptive). Shaltiel and Umans [36] showed that BPP_{path} is equal to $\text{P}_{||}^{\text{NP}}$ if $\text{E}_{||}^{\text{NP}}$ is hard for exponential size nondeterministic circuits.

Theorem 1.14 allows us to give a deterministic simulation of BPP_{path} algorithms with running time depending on the parameter $\epsilon(n)$. More specifically, it follows that under the hardness assumption, every BPP_{path} algorithm that gives an answer with probability $\epsilon(n)$, can be simulated in deterministic time $\text{poly}(n)/\epsilon(n)^2$.

---

[6] The assumption in Theorem 1.14 can be improved to E is hard for exponential size nondeterministic circuit. This is because Shaltiel and Umans [36] showed that this assumption implies that E is hard for exponential size Σ_1-circuits which make nonadaptive queries to their oracle. This latter assumption is sufficient for our proof. We defer the details to the final version.

1.11 PRGs with relative error

Theorem 1.14 demonstrates that it is sometimes possible to achieve consequences of PRGs with low error, under hardness assumptions that do not seem to suffice for such PRGs. We now give another such example.

A useful property of a δ-PRG $G : \{0,1\}^r \to \{0,1\}^n$ with very small error $\delta = n^{-\omega(1)}$ is that it "preserves the probability of small events". By that we mean that for every circuit $C : \{0,1\}^n \to \{0,1\}$ such that $\Pr[C(U_n) = 1] \leq \delta$,

$$\Pr[C(G(U_r)) = 1] \leq \Pr[C(U_n) = 1] + \delta \leq 2\delta$$

which is still negligible. The notion of PRGs with "relative error" defined below captures this property. In the definition below, the reader should think of $\delta \ll \epsilon$, and recall $e^\epsilon \approx 1 + \epsilon$ for sufficiently small ϵ.

▶ **Definition 1.15** (re-PRGs)**.** Let p_1, p_2 be two numbers, we define a relation on p_1, p_2 by:

$$p_1 \overset{re}{\sim}_{(\epsilon,\delta)} p_2 \;\Leftrightarrow\; \max(p_1, p_2) \leq e^\epsilon \cdot \min(p_1, p_2) + \delta.$$

A function $G : \{0,1\}^r \to \{0,1\}^n$ is an (ϵ, δ)-re-PRG for a class $\mathcal{C}$ of functions $C : \{0,1\}^n \to \{0,1\}$ if for every C in the class $\mathcal{C}$,

$$\Pr[C(G(U_r)) = 1] \overset{re}{\sim}_{(\epsilon,\delta)} \Pr[C(U_n) = 1].$$

The use of the formalism above is inspired by the notion of (ϵ, δ)-differential privacy. An $(1, \delta)$-re-PRG indeed "preserves the probability of small events" and gives that $\Pr[C(U_n) = 1] \leq \delta$ implies $\Pr[C(G(U_r)) = 1] \leq e \cdot \delta$. It also immediately follows that:

▶ **Fact 1.16.** *If G is an (ϵ, δ)-re-PRG for $\mathcal{C}$ and $\delta \leq \epsilon$ then G is a 4ϵ-PRG for $\mathcal{C}$.*

Thus, an (ϵ, δ)-re-PRG with $\delta \ll \epsilon$ can be thought of as an ϵ-PRG which has the additional property that it preserves the probability of small events.

Our next result is a construction of poly(n)-time computable (ϵ, δ)-re-PRGs with arbitrary polynomial stretch, $\epsilon = n^{-O(1)}$ and exponentially small $\delta = 2^{-\sqrt{r}} = 2^{-n^{\Omega(1)}}$.

▶ **Theorem 1.17.** *If E is hard for exponential size Σ_3-circuits, then for every constants $b, e > 1$ and $\mu > 0$ there exists a constant $\gamma > 0$ such that and every sufficiently large n, there is a function $G : \{0,1\}^{r=n^\mu} \to \{0,1\}^n$ that is an $(n^{-b}, 2^{-\gamma\cdot\sqrt{r}})$-re-PRG for size n^b circuits. Furthermore, G is computable in time poly(n^b).*

We remark that we would have liked to achieve $\delta = 2^{-\Omega(r)}$ (rather than $\delta = 2^{-\Omega(\sqrt{r})}$) but we don't know how to achieve this.

1.12 Randomness reduction in Monte-Carlo constructions

In many famous explicit construction problems (such as constructing rigid matrices or generator matrices for linear codes matching the Gilbert-Varshamov bound) a random n bit string has the required property with overwhelming probability of $1 - \delta$ for exponentially small δ. It is often the case that we do not have poly(n)-time deterministic algorithm that produce an n bit string with the required property. An intermediate goal is to reduce the number of random bits used (while preserving exponentially small failure probability). Using re-PRGs, achieves this task for problems where checking whether a given an n bit string x satisfies the property can be decided in the polynomial time hierarchy (and note that the two aforementioned construction problems satisfy this requirement). This is stated formally below:

▶ **Theorem 1.18.** *Let $i \geq 0$ be a constant and let L be a language such that:*

- *There is a constant $\alpha > 0$ s.t. for every sufficiently large n, $\Pr_{X \leftarrow U_n}[X \in L] \geq 1 - \delta$ for $\delta = 2^{-n^\alpha}$.*
- *L is accepted by a family of poly-size Σ_i-circuits.*

If E is hard for exponential size Σ_{i+3}-circuits then there is a poly(n)-time algorithm B such that $\Pr[B(U_r) \in L] \geq 1 - 4 \cdot \delta$ with $r = O(\log(1/\delta)^2) = n^{2\alpha}$.

Note that standard PRGs (which under this assumption achieve error $\geq 1/\text{poly}(n)$) give a version of Theorem 1.18 with $\delta = 1/\text{poly}(n)$. Our result gives this tradeoff also for smaller values of δ. We remark that we would have liked the dependence of r on δ in Theorem 1.18 to be $r = O(\log(1/\delta))$, and this would follow if we can improve the parameter $\delta = 2^{-\Omega(\sqrt{r})}$ in Theorem 1.17 to $\delta = 2^{-\Omega(r)}$.

The aforementioned examples of matrix rigidity and linear codes matching the Gilbert-Varshamov bound do not seem related to computational hardness assumptions. Assuming circuit lower bounds in order to handle them, may seem like an overkill. We remark that one can also apply Theorem 1.18 to solve explicit construction problem that are computational in nature. For example, the language L consisting of truth tables of functions $f : \{0,1\}^{\log n} \to \{0,1\}$ with almost maximal circuit complexity also satisfies the requirements in Theorem 1.18, and so, if E is hard for exponential size Σ_4-circuits, then there is a randomized polynomial time algorithm that uses $r = n^{2\alpha}$ random bits and generates an n-bit truth table of a function with almost maximal circuit complexity with probability at least $1 - 2^{-n^\alpha}$.

This approach can be useful to construct other "computational" pseudorandom objects, and is used to explicitly construct nonboolean PRGs (formally defined in the next section) with relative error, under hardness assumptions. This result is described in the next section.

1.13 PRGs with relative error for nonboolean distinguishers

Dubrov and Ishai [11] considered a generalization of PRGs which fools circuits that output many bits (and not just boolean circuits).

▶ **Definition 1.19** (nb-PRG)**.** Let ℓ be a parameter, and let $\mathcal{C}$ be a class of functions $C : \{0,1\}^n \to \{0,1\}^\ell$. A function $G : \{0,1\}^r \to \{0,1\}^n$ is an (ℓ, ϵ)-nb-PRG for $\mathcal{C}$ if for every C in $\mathcal{C}$, the probability distributions $C(G(U_r))$ and $C(U_n)$ are ϵ-close, meaning that for every function $D : \{0,1\}^\ell \to \{0,1\}$, $|\Pr[D(C(G(U_r))) = 1] - \Pr[D(C(U_n)) = 1]| \leq \epsilon$.

For every $\ell \geq 1$, an (ℓ, ϵ)-nb-PRG is in particular a $(1, \epsilon)$-nb-PRG which is easily seen to be equivalent to an ϵ-PRG. Thus, (ℓ, ϵ)-nb-PRGs are a generalization of ϵ-PRGs, and so the limitations of Theorem 1.12 apply to them.

The motivation for nb-PRGs is reducing the randomness complexity of sampling procedures. We now explain this application. Let P be a distribution over ℓ-bit strings, and let A be a sampling algorithm for it. That is, A is a poly(n)-time algorithm such that $A(U_n) = P$. An (ℓ, ϵ)-nb-PRG $G : \{0,1\}^r \to \{0,1\}^n$ for size n^b-circuits can be used to sample a distribution P' that is ϵ-close to P, using only $r < n$ random bits. This is because the sampling algorithm $B(U_r) = A(G(U_r))$ produces a distribution that is ϵ-close to $A(U_n)$.[7] Note that if we want B to run in time poly(n), we must require that G runs in time poly(n). Thus, this is another setting where we would like to have PRGs computable in time poly(n).

[7] It is important to note that if G is a standard PRG, we can only guarantee that $B(U_r)$ is computationally indistinguishable from $A(U_n)$, rather than statistically indistinguishable.

In this paper we consider a generalization of nb-PRGs with (ϵ, δ)-relative error.

▶ **Definition 1.20** (re-nb-PRG)**.** Let $\mathcal{C}$ be a class of functions $C : \{0,1\}^n \to \{0,1\}^\ell$. A function $G : \{0,1\}^r \to \{0,1\}^n$ is an (ℓ, ϵ, δ)-re-nb-PRG for $\mathcal{C}$ if for every C in $\mathcal{C}$, the probability distributions $C(G(U_r))$ and $C(U_n)$ are (ϵ, δ)-close in relative distance, meaning that for every function $D : \{0,1\}^\ell \to \{0,1\}$, $\Pr[D(C(G(U_r))) = 1] \overset{re}{\sim}_{(\epsilon,\delta)} \Pr[D(C(U_n)) = 1]$.

If we use re-nb-PRGs (rather than nb-PRGs) in the construction of the sampling algorithm B, then we "preserve probability of small events". That is for every $D : \{0,1\}^\ell \to \{0,1\}$, if $\Pr[D(P) = 1] \leq \delta$ then $\Pr[D(P') = 1] \leq O(\delta)$. This property is helpful in some applications.

Previous work by Applebaum et al. [2] (improving upon [11, 4]) gives $(\ell, n^{-O(1)})$-nb-PRGs with seed length $r = O(\ell + \log n)$ and $\epsilon = n^{-O(1)}$. This is under the assumption that E is hard for exponential size nondeterministic circuits. Note that $r \geq \ell$ is a trivial lower bound on the seed length. In this paper we construct (ϵ, δ)-re-nb-PRGs with $\epsilon = n^{-O(1)}$ and $r = 1 \cdot \ell + O(\log(1/\delta))^2$ for $\delta \geq 2^{-n^{\Omega(1)}}$. This is done under the stronger assumption that E is hard for exponential size Σ_6-circuits.

▶ **Theorem 1.21** (re-nb-PRG with seed length $1 \cdot \ell + O(\log(1/\delta))^2$)**.** *If E is hard for exponential size Σ_6-circuits then for every constants $b > 1$, $\alpha > 0$ there exists a constant $c > 1$ such that for every functions $\ell = \ell(n) \leq n$, $\delta = \delta(n) \leq 2^{-n^\alpha}$, and every sufficiently large n, there is a function $G : \{0,1\}^r \to \{0,1\}^n$ that is an (ℓ, ϵ, δ)-re-nb-PRG for circuits of size n^b with $\epsilon = n^{-b}$, and $r = \ell + c \cdot (\log(1/\delta))^2$.*

Note that the dependence of r on ℓ is an additive term of $1 \cdot \ell$. This is best possible, with the correct leading constant. We would have liked the dependence of r on δ to be an additive term of $O(\log(1/\delta))$. Once again, we would get this if we could improve the parameter $\delta = 2^{-\Omega(\sqrt{r})}$ in Theorem 1.17 to $\delta = 2^{-\Omega(r)}$. We also remark that the requirement that $\delta \leq 2^{-n^\alpha}$ may be omitted, and then $r = \ell + c \cdot ((\log(1/\delta))^2 + \log n)$.

1.14 Cryptographic applications of re-nb-PRGs

Dubrov and Ishai [11] observe that nb-PRGs can be used to reduce the randomness complexity of parties in multi-party cryptographic protocols. They consider the setup where honest parties run in polynomial time, and security is information theoretic (that is security is guaranteed even against unbounded adversaries). The precise details can be found in [11]. When using nb-PRGs, this application requires nb-PRGs with small error, as the probability of a security breach in the final protocol is additive in the error of the nb-PRG. However, assuming the probability of a security breach in the original protocol is at most δ (for some negligible δ), we can use $(\ell, 1, \delta)$-re-nb-PRGs to "preserve the probability of small events" and obtain a protocol with reduced randomness complexity, and where the probability of a security breach is at most $4 \cdot \delta$.

The key idea in the application above is that the nb-PRG is used to fool "honest parties" rather than "adversaries". This observation is crucial if we want to use NW-style PRGs in cryptography. More precisely, unlike "cryptographic PRGs" which fool circuits of superpolynomial size, NW-style PRGs (such as our re-nb-PRGs) only fool circuits of fixed polynomial size n^b and run in time $\text{poly}(n^b)$. Thus, they are unsuitable to fool cryptographic adversaries (which are more powerful than honest parties).

It is our hope that re-nb-PRGs may find other applications in cryptography. Toward this goal, we present the following toy example of a potential application of re-nb-PRGs: Suppose we are given a one-way function $f : \{0,1\}^{n^3} \to \{0,1\}^n$ that is computable in time n^b and circuits of very large size (say $s = 2^{n^{1/3}}$) cannot invert with probability larger than δ. Can

we reduce the input length of f to say $O(n)$ bits while preserving its security? Note that this only makes sense if we use tools that don't imply a stronger one-way function. We are not aware of such a conversion.

Nevertheless, using a poly(n)-time computable $(n, 1, \delta)$-re-nb-PRG $G : \{0,1\}^{O(n)} \to \{0,1\}^{n^3}$ for size n^b circuits (which we can achieve for $\delta = 2^{-\sqrt{n}}$ under Theorem 1.21) we can argue that $f'(x) = f(G(x))$ is a one-way function where the input length is reduced from n^3 to $O(n)$, and the security of f is preserved: circuits of size s can invert f' with probability at most $4 \cdot \delta$.

2 Overview of the technique

In this section we give a high level overview of the technique used to prove our results.

2.1 HSGs with low error

We assume that E is hard for exponential size nondeterministic circuits, and construct a poly(n)-time computable ϵ-HSG $G : \{0,1\}^{O(\log n)+\log(1/\epsilon)} \to \{0,1\}^n$ for circuits of fixed polynomial size. By Theorem 1.7 our assumption implies a poly(n)-time computable $\frac{1}{2}$-HSG $G' : \{0,1\}^{O(\log n)} \to \{0,1\}^{2n}$ for nondeterministic circuits of fixed polynomial size. It is standard that using $z \leftarrow U_{2n}$, we can produce $t = O(1/\epsilon) \leq 2^n$ pairwise independent random variables $Y_1(z), \ldots, Y_t(z)$ of length n. Furthermore, even though t may be super-polynomial, there is a polynomial time algorithm that given z, i, outputs the i'th variable $Y_i(z)$.

Our generator G will receive two seeds: a seed x for G', and an $i \in [t]$. It uses x to prepare a $2n$ bit long output string $z = G'(x)$, and then uses z as a seed to generate the i'th random variable $Y_i(z)$.

Let $D : \{0,1\}^n \to \{0,1\}$ be some fixed polynomial size deterministic circuit with $\Pr[D(U_n) = 1] \geq \epsilon$, and let $B = \{x : D(x) = 1\}$. By Chebyshev's inequality, pairwise independent variables have a "hitting property" for sets B of size at least $\epsilon \cdot 2^n$, meaning that with probability at least $2/3$ over choosing $z \leftarrow U_{2n}$, there exists an $i \in [t]$ such that $Y_i(z) \in B$ which means that $D(Y_i(z)) = 1$. Consider the nondeterministic circuit $C : \{0,1\}^{2n} \to \{0,1\}$, which given $z \in \{0,1\}^{2n}$ accepts iff $\exists i : D(Y_i(z)) = 1$. This is a fixed polynomial size nondeterministic circuit (and jumping ahead we mention that it uses $\log t = \log(1/\epsilon) + O(1)$ nondeterministic bits). We have that $\Pr_{z \leftarrow U_{2n}}[C(z) = 1] \geq 2/3$. Thus by the guarantee on G', there exists a seed x for G' such that $C(G'(x)) = 1$. This in turn means that there exists a seed (x, i) for G, such that $D(G(x, i)) = 1$ as required. The precise argument is given in Section 5.

The proof above uses the standard pairwise independent based randomness efficient amplification of success probability of randomized algorithms with a twist: The circuit C uses its nondeterminism to "speed up" the amplification as it does not have to explicitly go over all t options for i. A technically related (though somewhat different) idea was used by Paturi and Pudlak [30] in the context of "boosting" the success probability of hypothetical efficient randomized circuit-sat algorithms. There, given a circuit D, one considers a deterministic circuit D' which is hardwired with a "good string" z, and on input i, applies C on $Y_i(z)$. The key idea is that the input length of D' is $\log(1/\epsilon) < n$, and this is used to argue that feeding D' (rather than D) to the hypothetical circuit-sat algorithm, allows one to make progress.

2.2 Derandomization of randomized algorithms with large error

By going over all seeds of our ϵ-HSG we can derandomize one-sided error polynomial time algorithms with success probability ϵ and prove Theorem 1.9. We now explain that this

argument extends also to two-sided error algorithms of the form of Theorem 1.14. We make slight modifications in the construction above. This time we require that G' is a $\frac{1}{10}$-PRG for Σ_1-circuits, and by Theorem 1.7 such PRGs follow from the assumption that E is hard for exponential size Σ_1-circuits (and a more careful analysis allows an even weaker assumption). We also increase the number of pairwise independent variables from $O(1/\epsilon)$ to $t = O(1/\epsilon^2)$. We do this, as with this "query complexity", pairwise independent variables give an "averaging sampler", which means that for any set $B \subseteq \{0,1\}^n$, the fraction of Y_i's that land in B is with probability 9/10 close to the volume $\frac{|B|}{2^n}$ of B. Let G be the function obtained by these modifications.

We do not expect to prove that G is an ϵ-PRG, as by Theorem 1.12 such a proof will not be black-box. More concretely, the generator G' has an error of 1/10, and so a 1/10-fraction of its seeds may be useless, and we cannot hope that G has error $< 1/10$.

Let $D : \{0,1\}^n \to \{0,1\}$ be a fixed polynomial size circuit. In the two sided error case, we want to distinguish the case that $\Pr[D(U_n) = 1] \geq 2\epsilon$ from the case that $\Pr[D(U_n) = 1] \leq \epsilon$. We show how to use G in order to distinguish these two cases, in deterministic time $\text{poly}(n)/\epsilon^2$.

In analogy to the previous argument, we can show that with probability 9/10 over $z \leftarrow U_{2n}$, the estimate $p(z) = \frac{1}{t} \cdot |\{i : D(Y_i(z)) = 1\}|$ is very close to the acceptance probability of D. For simplicity, let us cheat and assume equality. The key observation is that by the classical results of [40, 39, 26] on approximate counting of NP witnesses (see Section 4.1 for a precise statement), $p(z)$ can be estimated by a fixed polynomial size Σ_1-circuit $C(z)$. Furthermore, this estimation is sufficiently accurate to distinguish the case that $p(z) \geq 2\epsilon$ from the case that $p(z) \leq \epsilon$. Similarly to the earlier argument, the fact that G' fools C, means that replacing $z \leftarrow U_{2n}$ with $G'(x) : x \leftarrow U_{O(\log n)}$ makes little difference. This means that by going over all $x \in \{0,1\}^{c \log n}$, and checking if for at least half of them $p(G'(x)) \geq 2\epsilon$, we can indeed distinguish the two cases. This takes time $\text{poly}(n)/\epsilon^2$ as required. The precise argument is given in Section 5.

2.3 HSGs with low error imply hardness for weak nondeterministic circuits

Let $G : \{0,1\}^r \to \{0,1\}^n$ be an ϵ-HSG for fixed polynomial size circuits. Note that in Section 2.1 we explained that such HSGs follow from $\frac{1}{2}$-HSGs for fixed polynomial size nondeterministic circuits with roughly $\log(1/\epsilon)$ nondeterministic bits. We now show that G implies such HSGs. Indeed consider, the function G' that outputs the first $n-k$ bits of the output of G, for $k = \log(1/\epsilon) - 1$. We show that G' is a $\frac{1}{2}$-HSG for fixed polynomial size circuits with k nondeterministic bits. Indeed, let $C : \{0,1\}^{n-k} \to \{0,1\}$ be such a circuit that accepts at least half of its inputs. This means that there exists a fixed polynomial size deterministic circuit $D : \{0,1\}^n \to \{0,1\}$, such that $C(x) = 1 \Leftrightarrow \exists y \in \{0,1\}^k$ s.t. $D(x,y) = 1$. The fact that C accepts half of its input implies that D accepts at least $\frac{1}{2} \cdot 2^{-k} = \epsilon$ fraction of the pairs $(x,y) \in \{0,1\}^n$, which implies that there exists a seed s for G such that $D(G(s)) = 1$. This in turn implies that $C(G'(s)) = 1$ as required.

There is an easy general transformation by Impagliazzo, Shaltiel and Wigderson [23] which transforms an HSG into a worst-case hard function in E. This transformation can be used to transform G' into a function that is hard for nondeterministic circuits with few nondeterministic bits. The precise argument is given in Section 6.

2.4 A construction of re-PRGs

Our starting point is a construction of Trevisan and Vadhan [43], which under the assumption that E is hard for exponential size Σ_1-circuits, gives a polynomial time computable function $f : \{0,1\}^n \to \{0,1\}^{n'=\Omega(n)}$ such that for every fixed polynomial size circuit $A : \{0,1\}^n \to \{0,1\}$, $\Pr_{V \leftarrow U_n}[A(V) = f(V)] \leq 2^{-n'/3}$.[8]

A natural approach toward constructing PRGs is to use the Goldreich-Levin theorem [14] to transform f into a *boolean* function g. Indeed, the standard way to do this is to define $g(v,y) = EC(f(v))_y$ where EC is a binary list-decodable code (and the GL theorem is for the special case of the Hadamard code). For this to work, we require that EC has an efficient list-decoding algorithm that can recover from distance $\frac{1}{2} - \epsilon$. In our setting, $\epsilon = 2^{-\Omega(n)}$, and we want a list-decoding algorithm implementable by a polynomial size circuit D. This is obviously impossible, as D needs to read at least $1/\epsilon$ positions in the "received word".

We may hope to circumvent this problem by allowing D to be a poly-size Σ_i-circuit. This will allow D to query the received word in exponentially many positions (on different computation paths). On the one hand, if we assume that E is hard for exponential size Σ_{i+1}-circuits, then the proof of Trevisan and Vadhan (which relativizes) gives security against Σ_i-circuits. However, the lower bounds of Applebaum et al. [2] show that even Σ_i-circuits cannot be used for this task of list decoding binary codes. Indeed, if we could get a boolean function g that cannot be computed with advantage better than $\epsilon = 2^{-\Omega(n)}$ over random guessing, we would obtain an $O(\epsilon)$-PRG by plugging g into the NW generator.

Instead, we shoot for a weaker conclusion, and try to show that the function g has the property that $G(x) = (x, g(x))$ is a $(2^{-\Omega(n)}, n^{-O(1)})$-re-PRG with one bit stretch. (Later, we will be able to get arbitrary stretch by plugging g in the NW-generator). That is, that for every fixed polynomial size circuit C, $\Pr[C(G(U_n)) = 1] \overset{re}{\sim}_{(n^{-O(1)},\delta)} \Pr[C(U_{n+1}) = 1]$ for $\delta = 2^{-\Omega(n)}$. We give a "list-decoding algorithm", that given C, constructs a Σ_2-circuit A that computes the function f too well. This allows us to choose $i = 2$ and start from the assumption that E is hard for exponential-size Σ_3-circuits.

Our "list-decoding algorithm" builds on ideas by Trevisan and Vadhan [43] and Applebaum et al. [2]. For our purposes it is more intuitive to restate the construction of g in an equivalent way: We set $g(v,y) = E(f(v), y)$ where E is a strong extractor with error $\delta^{O(1)}$, which allows seed length and entropy threshold of $O(\log(1/\delta))$.

After a suitable averaging argument, we get that for a non-negligible fraction of good v, a circuit C that distinguishes $G(U_n)$ from U_{n+1}, can be used to distinguish $(Y, E(z^*, Y))$ from uniform for $z^* = f(v)$. The guarantee of strong extractors says that there cannot be more than $poly(1/\delta)$ strings $z \in \{0,1\}^{n'}$ for which this distinguishing is possible. (As the uniform distribution over these z's would be a source on which the extractor fails).

The key observation is that we can design a Σ_1-circuit $B_v(z)$ which uses approximate counting of NP witnesses and accept iff C distinguishes $(Y, E(z,Y))$ from uniform with relative distance. This is because we can use approximate counting to estimate the acceptance probability of C on these two distributions.[9] We have that $z^* = f(v)$ is one of the few z's that B_v accepts. We can guess $z^* = f(v)$ by using random sampling of NP-witnesses [26, 7] to uniformly sample an accepting input of B_v. This strategy can be seen as a Σ_2-circuit A that given v computes $f(v)$ with probability $\delta^{O(1)} = 2^{-\Omega(n)}$, contradicting the hardness of f.

[8] This is another example showing that nondeterministic reductions can achieve very low error.

[9] It is important to note that here we critically use the fact that C distinguishes with relative distance, and we cannot hope to do this for an additive distance of $2^{-\Omega(n)}$. This is the reason why constructing re-PRGs with small δ is easier than constructing δ-PRGs.

We obtain re-PRGs with polynomial stretch by plugging g into the NW-generator. The analysis of the NW-generator can be used (with suitable modifications) to argue that if $G(x) = (x, g(x))$ is an re-PRG then we obtain an re-PRG with larger output with closely related ϵ, δ. An inherent limitation of the NW-generator (that is discussed in detail in [24]) gives that the seed length is quadratic in the input length of g. This is the reason why we get that the seed length has quadratic dependence on $\log(1/\delta)$. The precise argument is given in Section 7.

2.5 A construction of re-nb-PRGs

We first show that an $(\epsilon, \delta \cdot 2^{-\ell})$-re-PRG for Σ_1-circuits is an $(\ell, O(\epsilon), O(\delta))$-re-nb-PRG for deterministic circuits. This implication appears in Section 8.

This means that our previous construction of re-PRGs can give re-nb-PRGs assuming E is hard for exponential size Σ_4-circuits. A disadvantage of this approach is that because of the quadratic loss mentioned above, we obtain seed length approximately $r = O(\ell + \log(1/\delta))^2$. Previous work on nb-PRGs [4, 2] already achieved seed length that is linear in ℓ which is optimal up to constants. We can obtain seed length $r = 1 \cdot \ell + O(\log(1/\delta))^2$. That is, we can remove the quadratic dependence on ℓ but not on $\log(1/\delta)$.

For this, we imitate an approach developed by Applebaum et al. [2], which we can now improve as we can use re-PRGs instead of standard PRGs. We first show that with probability $1 - \delta$, a random $\text{poly}(n^b)$-wise independent hash function $h : \{0,1\}^r \to \{0,1\}^n$ is an re-PRG for size n^b Σ_1-circuits, with excellent dependence of r on ϵ and δ. We then show that checking whether a given circuit h is not an re-PRG for Σ_1-circuits can be done by Σ_3-circuits. Loosely speaking, this is because a Σ_3-circuit can guess a Σ_1-circuit that is not fooled by h, and use approximate counting of NP-witnesses (which costs an NP-oracle) to check whether that circuit is not fooled by the given circuit. (Here again, it is crucial that the notion of distance is relative, so that approximate counting can be used).

Finally, we construct the re-nb-PRG G as follows: We use two seeds x_1 for h and x_2 for an $(n^{-O(1)}, \delta)$-re-PRG G' for Σ_3-circuits (that we have under the assumption that E is hard for exponential size Σ_6-circuits). G computes $G'(x_2)$ and use this to choose a hash function h from the family. The final output is $h(x_1)$.

We have that a random h from the family is an re-PRG for Σ_1-circuits with probability $1 - \delta$, and that Σ_3-circuits can check whether h is an re-PRG for Σ_1-circuits. As re-PRGs preserve the probability of small events, we conclude that with probability $1 - 4\delta$ over the choice of x_2 we obtain a hash function h that is an re-PRG for Σ_1-circuits (which we already showed is an re-nb-PRG for deterministic circuits). Therefore, G is an re-nb-PRG. The precise argument is given in Section 8.

3 Organization of the paper

In Section 4 we state the classical results on approximate counting and sampling of NP witnesses. We also define several notions of relative approximation and prove some useful lemmas regarding them. In Section 5 we construct HSGs with low error, and prove Theorem 1.8. We also show how to derandomize two sided error algorithms and prove Theorem 1.14. In Section 6 we show that HSGs with low error are essentially equivalent to $\frac{1}{2}$-HSGs for nondeterministic circuits with few nondeterministic bits. We also prove Theorem 1.11 and show that HSGs with low error imply lower bounds for nondeterministic circuits with few nondeterministic bits. In Section 7 we give our construction of re-PRGs. In Section 8 we show how to use re-PRGs in order to construct re-nb-PRGs.

4 Preliminaries

4.1 Approximate counting and uniform sampling of NP witnesses

We use the classical result on approximate counting and uniform sampling of NP-witnesses [40, 39, 26, 7], which we now state in a way that is convenient for our application.

▶ **Definition 4.1** (relative approximation)**.** We say that a number p is an ϵ-relative approximation to q if $e^{-\epsilon} \cdot q \leq p \leq e^{\epsilon} \cdot q$.

▶ **Theorem 4.2** (approximate counting [40, 39, 26])**.** *For every $i \geq 0$, every sufficiently large s and every $0 < \epsilon \leq 1$, there is a Σ_{i+1}-circuit of size poly(s/ϵ) that given a Σ_i-circuit C of size s outputs an ϵ-relative approximation of $|\{x : C(x) = 1\}|$.*

▶ **Theorem 4.3** (uniform sampling [26, 7])**.** *For every $i \geq 0$, every sufficiently large s and every $\delta > 0$, there is a randomized size poly$(s, \log(1/\delta))$ Σ_{i+1}-circuit A that given a Σ_i-circuit $C : \{0,1\}^n \to \{0,1\}$ of size $s \geq n$ outputs a value in $\{0,1\}^n \cup \{\perp\}$ such that for every size s Σ_i-circuit, $\Pr[A(C) = \perp] \leq \delta$ and the distribution $(A(C)|A(C) \neq \perp)$ is uniform over $\{x : C(x) = 1\}$.*

4.2 Notions of relative error

In Section 1 we defined a notion of relative distance between two numbers which we notate by $p_1 \overset{re}{\sim}_{(\epsilon,\delta)} p_2$. This notion was used in the definition of re-PRGs and re-nb-PRGs. In this section we discuss properties of this distance, as well as related notions of distance.

▶ **Definition 4.4.** Let p_1, p_2 be two numbers, and let $p_{\max} = max(p_1, p_2)$ and $p_{\min} = min(p_1, p_2)$. We say that p_1, p_2 are

- ϵ-**close** if $p_{\max} - p_{\min} \leq \epsilon$, and use the notation $p_1 \overset{ad}{\sim}_{\epsilon} p_2$.
- (ϵ, δ)-**relative-close** if $p_{\max} \leq e^{\epsilon} \cdot p_{\min} + \delta$, and use the notation $p_1 \overset{re}{\sim}_{(\epsilon,\delta)} p_2$.
- (ϵ, δ)-**relative-threshold-close** if $p_{\max} \leq \delta$ or $p_{\max} \leq e^{\epsilon} \cdot p_{\min}$, and use the notation $p_1 \overset{rt}{\sim}_{(\epsilon,\delta)} p_2$.

The three notions above can be used to define distance between probability distributions. Thus, for example, if X, Y are distributions over a finite set Ω, we write $X \overset{re}{\sim}_{(\epsilon,\delta)} Y$ if for every function $D : \Omega \to \{0,1\}$, $\Pr[D(X) = 1] \overset{re}{\sim}_{(\epsilon,\delta)} \Pr[D(Y) = 1]$.

It is easy to verify the following relationships between the three notions, by using the approximations $1 + x \leq e^x \leq 1 + 3x$ and $1 - x \leq e^{-x} \leq 1 - x/3$ which hold for $0 \leq x \leq 1$

▶ **Lemma 4.5.** *For every numbers $0 \leq p_1, p_2 \leq 1$, and $0 \leq \epsilon, \delta \leq 1$*

- $p_1 \overset{re}{\sim}_{(\epsilon,\delta)} p_2 \Rightarrow p_1 \overset{ad}{\sim}_{3\epsilon+\delta} p_2$.
- $p_1 \overset{rt}{\sim}_{(\epsilon,\delta)} p_2 \Rightarrow p_1 \overset{re}{\sim}_{(\epsilon,\delta)} p_2$.
- *For $\epsilon \leq \frac{1}{2}$,* $p_1 \overset{re}{\sim}_{(\epsilon,\delta)} p_2 \Rightarrow p_1 \overset{rt}{\sim}_{(4\epsilon,4\delta/\epsilon)} p_2$.
- $p_1 \overset{rt}{\sim}_{(\epsilon,\delta)} p_2 \Rightarrow p_1 \overset{ad}{\sim}_{3\epsilon+\delta} p_2$.
- $p_1 \overset{ad}{\sim}_{\delta} p_2 \Rightarrow p_1 \overset{rt}{\sim}_{(\epsilon,3\delta/\epsilon)} p_2$.

In our constructions of re-PRGs and re-nb-PRGs, we will shoot for $\epsilon = n^{-O(1)}$ and $\delta = 2^{-n^{\Omega(1)}}$. Note that by Lemma 4.5, for these choices, the notions of "relative-close" and "relative-threshold-close" are equivalent. It turns out that for our purposes, the notion of "relative-threshold-close" is easier to work with. For this reason we now redefine the notions of re-PRGs and re-nb-PRGs using the notion of relative-threshold-close instead

of relative-close. The definitions are identical except that we use "relative-threshold-close" instead of "relative-close".

▶ **Definition 4.6** (rt-PRGs). A function $G : \{0,1\}^r \to \{0,1\}^n$ is an (ϵ,ρ)-rt-PRG for a class $\mathcal{C}$ of functions $C : \{0,1\}^n \to \{0,1\}$ if for every C in the class $\mathcal{C}$,

$$\Pr[C(G(U_r)) = 1] \overset{rt}{\sim}_{(\epsilon,\rho)} \Pr[C(U_n) = 1].$$

▶ **Definition 4.7** (rt-nb-PRG). Let $\mathcal{C}$ be a class of boolean functions $C : \{0,1\}^n \to \{0,1\}^\ell$. A function $G : \{0,1\}^r \to \{0,1\}^n$ is an (ℓ,ϵ,ρ)-rt-nb-PRG for $\mathcal{C}$ if for every C in $\mathcal{C}$, the probability distributions $C(G(U_r)) \overset{rt}{\sim}_{(\epsilon,\rho)} C(U_n)$.

By the discussion above it immediately follows that:

▶ **Fact 4.8** (rt-PRGs are re-PRGs). *An (ϵ,ρ)-rt-PRG is also an (ϵ,ρ)-re-PRG, and an (ϵ,ρ)-rt-nb-PRG is also an (ϵ,ρ)-re-nb-PRG.*

In the remainder of the paper we will only discuss rt-PRGs.

4.3 Some useful technical lemmas on relative error

The next lemma shows that if we can approximate two quantities p_1, p_2 using an η-relative approximation, for $\eta < \epsilon/10$ then when we can essentially tell if $p_1 \overset{rt}{\not\sim}_{(\epsilon,\rho)} p_2$.

▶ **Lemma 4.9.** *Let $0 \le p_1, p_2 \le 1$ and let p'_1, p'_2 be η-relative approximations of p_1, p_2 respectively. Let $T(p'_1, p'_2)$ be a test that accepts iff $\max(p'_1, p'_2) \ge \rho \cdot e^{-\eta}$ and $\frac{\max(p'_1,p'_2)}{\min(p'_1,p'_2)} \ge e^{\epsilon-2\eta}$. Then,*

- $p_1 \overset{rt}{\not\sim}_{(\epsilon,\rho)} p_2 \Rightarrow T(p'_1, p'_2)$ *accepts.*
- $T(p'_1, p'_2)$ *accepts* $\Rightarrow p_1 \overset{rt}{\not\sim}_{(\epsilon-4\eta,\rho\cdot e^{-2\eta})} p_2$.

We also need the following technical lemma that allows us to perform "Markov style arguments" with relative distance.

▶ **Lemma 4.10.** *Let R, W be independent random variables, and let ψ_1, ψ_2 be boolean functions. Assume that*

$$\Pr[\psi_1(R,W) = 1] \overset{rt}{\not\sim}_{(\epsilon,\rho)} \Pr[\psi_2(R,W) = 1].$$

Let $\epsilon' = \epsilon/10$ and $\rho' = \rho \cdot \epsilon/10$, and let $G = \left\{ r : \Pr[\psi_1(r,W) = 1] \overset{rt}{\not\sim}_{(\epsilon',\rho')} \Pr[\psi_2(r,W) = 1] \right\}$. Then $\Pr[R \in G] \ge \rho \cdot \epsilon/10$.

Proof. Let $a_r = \Pr[\psi_1(r,W) = 1]$, $b_r = \Pr[\psi_2(r,W) = 1]$ and $p_r = \Pr[R = r]$. We can write $a = \Pr[\psi_1(R,W) = 1] = \sum_r p(r) a_r$ and $b = \Pr[\psi_2(R,W) = 1] = \sum_r p(r) b_r$. Assume w.l.o.g. that $a > b$ and we know that $a > e^\epsilon b$ and $a > \rho$. We conclude that $a - b = \sum_r p(r)(a_r - b_r) > \max((e^\epsilon - 1)b, \rho - b)$ and assume that $(a_r - b_r)$ is positive for all r (otherwise we take only positive terms and increase the sum).

Let $A = \{r : a_r > e^{\epsilon'} b_r \wedge a_r > \rho'\}$, $B = \{r : a_r \le \rho'\}$, $C = \{r : a_r \le e^{\epsilon'} b_r \wedge a_r > \rho'\}$.

$$\sum_{r\in A} p(r) \ge \sum_{r\in A} p(r)(a_r - b_r) > \max((e^\epsilon - 1)b, \rho - b) - \sum_{r\in B} p(r)(a_r - b_r) - \sum_{r\in C} p(r)(a_r - b_r).$$

Goal: Construct a poly(n)-time computable ϵ-HSG, $G : \{0,1\}^r \to \{0,1\}^n$ for circuits of size n^b with $r = 1 \cdot \log(1/\epsilon) + O(b \cdot \log(\cdot n))$.

Assumption: E is hard for exponential size nondeterministic circuits.

Parameters:

- b, n - We are shooting to fool circuits of size n^b.
- $\epsilon \geq 2^{-(n-1)}$ - the required error.

Ingredients:

- We make use of a $\frac{1}{2}$-HSG for nondeterministic circuits. Specifically, let $b' = a \cdot b$ for a constant $a > 1$ to be chosen later. By Theorem 1.7, there exists a constant $c > 1$ such that the assumption that E is hard for exponential size nondeterministic circuits, implies that for every sufficiently large n, there is a poly(n)-time computable $G' : \{0,1\}^{c \log n} \to \{0,1\}^{2n}$ that is a $\frac{1}{2}$-HSG for size $n^{b'}$ nondeterministic circuits.
- A hitter, that is a function hitter : $\{0,1\}^{2n} \to (\{0,1\}^n)^{4/\epsilon}$ such that for every $B \subseteq \{0,1\}^n$, $\Pr_{Z \leftarrow U_{2n}}[\exists i : \text{ hitter}(Z)_i \in B] \geq \frac{2}{3}$. It is standard that this is achieved by the "pair-wise independent hitter" that uses its $2n$ bit input to sample $4/\epsilon$ pairwise independent n bit variables, (see e.g., [13]). Moreover, given (z, i), hitter$(z)_i$ can be computed in time poly(n).

The HSG: Define $G : \{0,1\}^{c \log n + \log(4/\epsilon)} \to \{0,1\}^n$ by $G(x, i) = \text{hitter}(G'(x))_i$.

Figure 1 An HSG for low error.

Since $\sum_{r \in B} p(r)(a_r - b_r) \leq \rho'$ and $\sum_{r \in C} p(r)(a_r - b_r) \leq (e^{\epsilon'} - 1)b$ we conclude

$$\sum_{r \in A} p(r) > \max((e^{\epsilon} - e^{\epsilon'})b - \rho', \rho - e^{\epsilon'} b - \rho').$$

If $b < 0.25\rho$ then $\rho - e^{\epsilon'} b - \rho' > \rho(1 - e/4 - 1/10) > 0.22\rho$. If $b \geq 0.25\rho$ then $(e^{\epsilon} - e^{\epsilon'})b - \rho' > (0.9 \cdot 0.25 - 0.1)\epsilon\rho = 0.125\epsilon\rho$. So we can conclude that $\Pr[R \in G] = \sum_{r \in G} p(r) \geq \sum_{r \in A} p(r) > 0.125\epsilon\rho$ since $A \subseteq G$. ◀

5 Derandomization of poly-time randomized algorithms with large error

In this section we prove construct the low-error HSG of Theorem 1.8 and show how to extend the argument to handle two-sided error algorithms, proving Theorem 1.14.

5.1 An HSG for low error

We first consider the case of one-sided error algorithms which can be derandomized using hitting-set generators. The following theorem gives a construction of HSGs and implies Theorem 1.8 and Theorem 1.9.

▶ **Theorem 5.1** (HSG with seed length $\log(1/\epsilon) + O(\log n)$)**.** *Let $b > 1$ be a constant, and let $\epsilon = \epsilon(n) \geq 2^{-(n-1)}$. Assume that E is hard for exponential size nondeterministic circuits. Let G be the function constructed in Figure 1, with the parameters chosen there. Then there exists a constant $c > 1$ such that for every sufficiently large n, $G : \{0,1\}^{\log(1/\epsilon) + c \log n} \to \{0,1\}^n$ is an ϵ-HSG for size n^b circuits. Furthermore, G is computable in time poly(n^b).*

Proof. Let $D : \{0,1\}^n \to \{0,1\}$ be a size n^b circuit, let $B = \{y : D(y) = 1\}$, and assume that $|B| \geq \epsilon \cdot 2^n$. Let $T = \left\{z \in \{0,1\}^{2n} : \exists i : \text{ hitter}(z)_i \in B\right\}$. By the properties of hitter

$|T| \geq \frac{2}{3} \cdot 2^{2n}$. Note that by the definition of T, there exists a nondeterministic circuit $C : \{0,1\}^{2n} \to \{0,1\}$ of size $\text{poly}(n) + n^b$ such that $C(z) = 1$ iff $z \in T$. We can choose the constant a to be sufficiently large so that $n^{b'} = n^{a \cdot b}$ is larger than the size of C. It follows that G' fools C and in particular, there exists $x \in \{0,1\}^{c \log n}$ such that $C(G'(x)) = 1$ which implies $D(G(x,i)) = D(\text{hitter}(G'(x)_i) = 1$ as required. ◀

5.2 Extending the argument to 2-sided error randomized algorithms

In this section we prove Theorem 1.14. Our first step is to modify the construction in Figure 1 to use a PRG instead of an HSG and an averaging sampler instead of a hitter. Specifically, we replace hitter : $\{0,1\}^{2n} \to (\{0,1\}^n)^{4/\epsilon}$ with a function samp : $\{0,1\}^{2n} \to (\{0,1\}^n)^t$ for $t = O(1/\epsilon^2)$. that has the property that for every $B \subseteq \{0,1\}^n$ with $|B| \geq \frac{2}{3} \cdot 2^n$,

$$\Pr_{Z \leftarrow U_{2n}} \left[\left| \frac{|\{i : \text{samp}(Z)_i \in B\}|}{t} - \frac{|B|}{2^n} \right| \geq \frac{\epsilon}{10} \right] \leq \frac{1}{10}.$$

It is standard that this is achieved by the "pair-wise independent sampler" that uses its $2n$ bit input to sample $t = O(1/\epsilon)^2$ pairwise independent n bit variables, (see e.g., [13]). We also require that G' is a 1/10-PRG for Σ_1-circuits of size $n^{b'}$ (rather than a $\frac{1}{2}$-HSG for nondeterministic circuits of size $n^{b'}$). This follows just the same from Theorem 1.7, if we strengthen the assumption, and assume that E is hard for exponential size Σ_1-circuits. We repeat the construction of Figure 1 with these choices, and let G denote the final function G obtained in Figure 1 with the modifications above. We prove the following extension of Theorem 5.1.

▶ **Theorem 5.2.** *Let $b > 1$ be a constant, and let $\epsilon = \epsilon(n) \geq 2^{-n/2}$. Assume that E is hard for exponential size Σ_1-circuits. Let G be the function constructed in Figure 1, with the parameters chosen there and the modifications explained above. Then there exists a constant $c > 1$ such that for every sufficiently large n, $G : \{0,1\}^{c \log n} \times \{0,1\}^{\log t} \to \{0,1\}^n$ satisfies that for every size n^b circuit $D : \{0,1\}^n \to \{0,1\}$:*

- *If $\Pr[D(U_n) = 1] \geq 2 \cdot \epsilon$ then $\Pr_{X \leftarrow U_{c \log n}} \left[\frac{|\{i : D(G(X,i)) = 1\}|}{t} \geq \frac{3}{2} \cdot \epsilon \right] \geq \frac{4}{5}$.*
- *If $\Pr[D(U_n) = 1] \leq \epsilon$ then $\Pr_{X \leftarrow U_{c \log n}} \left[\frac{|\{i : D(G(X,i)) = 1\}|}{t} \geq \frac{3}{2} \cdot \epsilon \right] \leq \frac{1}{5}$.*

Furthermore, G is computable in time poly(n^b).

By the discussion in Section 1.8 we cannot use black-box techniques to construct poly-time computable PRGs with low error under the assumption of Theorem 5.2. Theorem 5.2 does not contradict the discussion as the constructed object G is not a PRG. It is not the case that $G(U_r)$ indistinguishable from uniform with very low error. Nevertheless, the guarantee on G suffices for derandomization in time exponential in the seed length (as is the case in PRGs).

Proof. (of Theorem 5.2) Let $D : \{0,1\}^n \to \{0,1\}$ be a size n^b circuit, let $B = \{y : D(y) = 1\}$. Let

$$T = \left\{ z \in \{0,1\}^{2n} : \left| \frac{|\{i : \text{samp}(z)_i \in B\}|}{t} - \frac{|B|}{2^n} \right| \leq \frac{\epsilon}{10} \right\}.$$

By the properties of samp, $|T| \geq \frac{9}{10} \cdot 2^{2n}$. It follows that:

- If $\Pr[D(U_n) = 1] \geq 2 \cdot \epsilon$ then $\Pr_{Z \leftarrow U_{2n}} \left[\frac{|\{i : D(\text{samp}(Z)_i) = 1\}|}{t} \geq \frac{7}{4} \cdot \epsilon \right] \geq \frac{9}{10}$.
- If $\Pr[D(U_n) = 1] \leq \epsilon$ then $\Pr_{Z \leftarrow U_{2n}} \left[\frac{|\{i : D(\text{samp}(Z)_i) = 1\}|}{t} \leq \frac{5}{4} \cdot \epsilon \right] \geq \frac{9}{10}$.

Consider the Σ_1-circuit $C : \{0,1\}^{2n} \to \{0,1\}$ that works as follows: given input $z \in \{0,1\}^{2n}$, C uses Theorem 4.2 to compute an η-relative approximation p' of $p = |\{i \in [t] : D(\mathrm{samp}(z)_i) = 1\}|$, for $\eta = 1/100$. The circuit C accepts iff $p' \geq t \cdot \frac{3}{2} \cdot \epsilon$. It follows that C is a Σ_1-circuit of size $\mathrm{poly}(n^b)$. The quality of approximation is sufficient to distinguish the case that $p \geq \frac{7}{4} \cdot \epsilon$ and $p \leq \frac{5}{4} \cdot \epsilon$.

We can choose the constant a to be sufficiently large so that $n^{b'} = n^{a \cdot b}$ is larger than the size of C. It follows that G' fools C, and the theorem follows.[10] ◀

We are now ready to prove Theorem 1.14.

Proof of Theorem 1.14. Let $T(n) \geq n$ be a bound on the running time of A. Given an input $x \in \{0,1\}^n$, we consider the circuit $D_x : \{0,1\}^{T(n)} \to \{0,1\}$, which given y simulates $A(x)$ using y as random coins. We apply Theorem 5.2 to obtain a constant c and a function

$$G : \{0,1\}^{c \log T(n)} \times \{0,1\}^{\log t(n)} \to \{0,1\}^{T(n)},$$

where $t(n) = O(1/\epsilon(n)^2)$. By applying the guarantee of Theorem 5.2 on D_x we get that

- $x \in L \Rightarrow \Pr[D_x(U_n) = 1] \geq 2 \cdot \epsilon] \Rightarrow \Pr_{S \leftarrow U_{c \log T(n)}} \left[\frac{|\{i : D_x(G(S,i)) = 1\}|}{t(n)} \geq \frac{3}{2} \cdot \epsilon(n) \right] \geq \frac{4}{5}$.
- $x \notin L \Rightarrow$ If $\Pr[D(U_n) = 1] \leq \epsilon] \Rightarrow \Pr_{S \leftarrow U_{c \log T(n)}} \left[\frac{|\{i : D_x(G(S,i)) = 1\}|}{t(n)} \geq \frac{3}{2} \cdot \epsilon(n) \right] \leq \frac{1}{5}$.

The deterministic algorithm works as follows: We go over all $s \in \{0,1\}^{c \log T(n)}$. For each one we count the number of $i \in [t(n)]$ for which D_x accepts $G(s,i)$. If the fraction of s, such that

$$\frac{|\{i : D_x(G(s,i)) = 1\}|}{t(n)} \geq \frac{3}{2} \cdot \epsilon(n)$$

is larger than $\frac{4}{5}$, we accept. The correctness of this simulation follows. The running time is $t(n) \cdot \mathrm{poly}(T(n)) = \mathrm{poly}(T(n))/\epsilon(n)^2$. ◀

6 On minimal hardness assumptions for HSGs with low error

We are using hardness for nondeterministic circuits in Theorem 1.8 which constructs an ϵ-HSG with very low error. Is this assumption necessary?

In this section we address this question. We will consider a natural intermediate model of circuits that are stronger than deterministic circuits, and weaker than nondeterministic circuits, namely nondeterministic circuits that use $k \leq n$ nondeterministic bits (as defined in Definition 1.4).

6.1 $\frac{1}{2}$-HSGs for nondeterministic circuits with few nondeterministic bits

An inspection of our construction of HSG in Figure 1 reveals that the assumption that E is hard for exponential size nondeterministic circuits was only used to obtain a $\frac{1}{2}$-HSG for nondeterministic circuits with a number of nondeterministic bits that is roughly $k = \log(1/\epsilon)$. More precisely, our construction gives the following general conversion.

[10] Note that the circuit C uses its oracle only to perform approximate counting. It can be verified that this can be done by a circuit C that makes nonadaptive queries to its oracle. This means that for this argument it is sufficient that G' fools circuits of this type, and by the "downward collapse theorem" of Shaltiel and Umans [36], coupled with Theorem 1.7, such PRGs follow under the seemingly weaker assumption that E is hard for exponential size nondeterministic circuits.

▶ **Theorem 6.1.** *For every constant $b > 1$ there is a constant $b' > b$ such that for every sufficiently large n, if $G : \{0,1\}^r \to \{0,1\}^{2n}$ is a $\frac{1}{2}$-HSG for size $n^{b'}$ nondeterministic circuits that use $k = \log(1/\epsilon) + O(1)$ nondeterministic bits, then there is a function $G' : \{0,1\}^{r+k} \to \{0,1\}^n$ that is an ϵ-HSG for circuits of size n^b. Furthermore, G can be computed in time poly(n) with one oracle call to G.*

We can show the following partial converse:

▶ **Lemma 6.2.** *Let $G : \{0,1\}^r \to \{0,1\}^n$ be an ϵ-HSG for circuits of size s, and let $G' : \{0,1\}^r \to \{0,1\}^{n-k}$ be $G'(x) = G(x)|_{1,\ldots,n-k}$ for $k = \log(1/\epsilon) - 1$. G' is a $\frac{1}{2}$-HSG for size s nondeterministic circuits with k nondeterministic bits.*

Proof. Let $C : \{0,1\}^{n-k} \to \{0,1\}$ be a size s nondeterministic circuit with k nondeterministic bits, which accepts at least half of its inputs. That is, there exists a deterministic circuit $D : \{0,1\}^{n-k} \times \{0,1\}^k \to \{0,1\}$ of size s such that: for every $x \in \{0,1\}^{n-k}$,

$$C(x) = 1 \;\Leftrightarrow\; \exists y \in \{0,1\}^k \text{ s.t. } D(x,y) = 1,$$

and $\Pr[C(U_{n-k}) = 1] \geq \frac{1}{2}$. It follows that $\Pr[D(U_n) = 1] \geq \frac{1}{2} \cdot 2^{-k} = \epsilon$ (here we view D as a circuit with n input bits). Thus, by the guarantee of the HSG, there exists an $s' \in \{0,1\}^r$ such that $D(G(s')) = 1$. Denote $G(s') = (x', y')$ so that $G'(s') = x'$. It follows that $D(x', y') = 1$ which implies that $C(x') = 1$, and we have that $C(G'(s')) = 1$. ◀

This means that in the case that $r = \Omega(\log(1/\epsilon))$ the notions of $\frac{1}{2}$-HSG for nondeterministic circuits with $k = \log(1/\epsilon)$ bits of nondeterminism, and the notion of ϵ-HSGs for standard circuits are essentially equivalent (in the sense that conversions between them incur only slight penalties in seed length and circuit size).

Consequently, if we are interested in low error HSGs for deterministic circuits that have polynomial stretch, we should be interested in HSGs against the class of polynomial size nondeterministic circuits on n bits with $\gamma \cdot n$ nondeterministic bits, for a small $\gamma > 0$.

6.2 Hardness assumptions that imply HSGs for circuits with weak nondeterminism

How hard is it to construct $\frac{1}{2}$-HSGs for poly-size nondeterministic circuits with $\gamma \cdot n$ nondeterministic bits? Given the success of the hardness versus randomness paradigm exhibited in Theorems 1.3 and Theorem 1.7, we can hope that hardness for this circuit class, translates into pseudorandomness for this circuit class. If this is the case, we can start from the assumption that there exists a $\gamma > 0$ such that E is hard for exponential size nondeterministic circuits that use γn nondeterministic bits.

An inspection of the hardness versus randomness tradeoffs in the literature reveal that they do not give such a result. Loosely speaking, because of the need for a hybrid argument, the reductions need to "perform decoding" from error less than $1/n$ and make a super-linear number queries to the distinguisher circuit. This overall means that we require hardness against circuits with a super-linear number of nondeterministic bits.

This is a pity because it trivially follows that nondeterministic circuits of size s with k nondeterministic bits can be simulated by deterministic circuits of size $s \cdot 2^k$, and this implies that:

▶ **Fact 6.3.** *If E is hard for exponential size circuits, then there exists a $\gamma > 0$ such that E is hard for exponential size nondeterministic circuits that use γn nondeterministic bits.*

Consequently, a hardness versus randomness tradeoff of the form we hope for, would be able to start from the assumption that E is hard for exponential size circuits. By the discussion in the introduction, such hardness versus randomness tradeoffs cannot have black-box proofs.

6.3 Hardness assumptions implied by HSGs with low error

In the previous section we observed that it is unlikely that we can prove that hardness for nondeterministic circuits with few nondeterministic bits implies HSGs with low error against deterministic circuits. We are able to prove the other direction. Specifically, we show that HSGs with low error and polynomial stretch, that run in time exponential in their seed length, imply that E is hard for polynomial size nondeterministic circuits with $\Omega(n)$ nondeterministic bits on inputs of length n. The following is a restatement of Theorem 1.11

▶ **Theorem 6.4.** *Let $\delta > 0$ be a constant. Assume that for every sufficiently large n, there is a 2^{-n^δ}-HSG $H : \{0,1\}^{O(n^\delta)} \to \{0,1\}^n$ for size $s \geq n$ circuits, and furthermore that the family of functions $H = \{H_n\}$ is computable in time exponential in the seed length, that is time $2^{O(n^\delta)}$. Then, there exists a constant $\gamma > 0$, and a problem $L \in E$ such that for every sufficiently large n', nondeterministic circuits of size $(\gamma n')^{1/\delta}$ with $\gamma \cdot n'$ nondeterministic bits fail to compute the characteristic function of L on inputs of length n'.*

Our current state of knowledge doesn't give us any reason to think that HSGs with $\epsilon = 1/n$ imply the same conclusion.

We use the following simple argument from [23] to prove the following:

▶ **Theorem 6.5.** *Let $H : \{0,1\}^r \to \{0,1\}^n$ be a $\frac{1}{2}$-HSG for size s nondeterministic circuits that use k bits. Let $f : \{0,1\}^{r+2} \to \{0,1\}$ be the function*

$$f(y) = 0 \;\Leftrightarrow\; \exists z \in \{0,1\}^{n-(r+2)}, \exists x \in \{0,1\}^n \text{ s.t. } H(x) = y \circ z$$

where "$\circ$" denotes concatenation. f cannot be computed by size s-circuits that use k bits of nondeterminism.

Proof. A circuit $C : \{0,1\}^{r+2} \to \{0,1\}$ computing f, can be thought of a circuit $C : \{0,1\}^n \to \{0,1\}$ (that only looks at the $r+2$ prefix of its input). It is immediate that $\Pr[C(U_n) = 1] \geq \frac{3}{4}$ and yet C answers zero on all outputs of G. ◀

Theorem 6.4 now follows by converting the low error HSG into a $\frac{1}{2}$-HSG for nondeterministic circuits with few nondeterministic bits, and then using Theorem 6.5 to convert the HSG into a hard function.

Proof of Theorem 6.4. Let c be the constant hidden in the seed length of H, and let n be sufficiently large. By Lemma 6.2 we have that $H' : \{0,1\}^r \to \{0,1\}^{n-k}$ is a $\frac{1}{2}$-HSG for size $s \geq n$, nondeterministic circuits with k bits of nondeterminism, for $k = n^\delta - 1$. Let $n' = c \cdot n^\delta + 2$, and let $f : \{0,1\}^{n'} \to \{0,1\}$ be the function obtained by applying Theorem 6.5 on H'. We have that f cannot be computed by size s-circuits that use k bits of nondeterminism. Note that $s \geq n \geq \Omega(n')^{1/\delta}$, and $k \geq \Omega(n')$. Let L be the language of the decision problem f. It follows that $L \in \mathrm{E}$ as f can be computed by running over all $2^{O(n^\delta)}$ seeds of G and computing G, and this takes time $2^{O(n^\delta)} = 2^{O(n')}$. ◀

7 A construction of an re-PRG

In this section we construct re-PRGs and prove Theorem 1.17. We will actually construct rt-PRGs which are in particular re-PRGs. We begin by constructing a poly(n)-time computable function $g : \{0,1\}^n \to \{0,1\}$, such that the function $G : \{0,1\}^n \to \{0,1\}^{n+1}$ defined by $G(x) = (x, g(x))$ is an $(1/\text{poly}(n), 2^{-\Omega(n)})$-rt-PRG for fixed polynomial size circuits. This construction is given in Section 7.1 and builds on ideas from [43, 2]. In order to obtain an re-PRG with polynomial stretch we apply the Nisan-Wigderson generator [29] where the function g plays the role of the hard function. The analysis of the NW-generator can be used (with some modifications) to argue that this yields an rt-PRG.

7.1 rt-PRGs with one bit stretch

7.1.1 The construction

We use the following result by Trevisan and Vadhan [43].

▶ **Theorem 7.1** ([43]). *Let $i \geq 0$. If E is hard for exponential size Σ_{i+1}-circuits, then for every constant $b > 1$, there exists some constant $\alpha > 0$ such that for every sufficiently large n, there is a function $f : \{0,1\}^n \to \{0,1\}^{m=\alpha \cdot n}$ such that for every size n^b, Σ_i-circuit A, $\Pr_{X \leftarrow U_n}[A(X) = f(X)] \leq 2^{-m/3}$. Furthermore, f is computable in time poly(n^b).*

▶ Remark. Theorem 7.1 is not stated in this form in [43]. Nevertheless, it is implicit in the work of [43] as we now explain.

Lemma 5.1 in [43] states that if a circuit C computes a degree d multivariate polynomial $p : \mathbb{F}^t \to \mathbb{F}$ (over a field $\mathbb{F}$ of size q) correctly on an ϵ fraction of its inputs (and if certain conditions on the parameters t, d, q and ϵ are met) then there exists a Σ_j-circuit C' that computes p correctly on all inputs, and the size of C' is polynomial in the size of C and in $t, d, \log q$ (but does not depend on ϵ). The lemma claims this for $j = 2$, but we will later observe that this holds also for $j = 1$.

The parameters in the lemma allow ϵ which is roughly $\sqrt{d/q}$ and thinking of p as a boolean function outputting $\log q$ bits, this allows ϵ to approach $2^{-\frac{1}{2} \cdot \log q}$ if $d \ll q$.

By using the "low degree extension" it is standard that E has complete problems that can be represented as such low degree polynomials. More precisely, given an input length n, we consider a restriction of an E complete problem to inputs of length $\ell = c \log n$ and perform the low degree extension with $d = 2^{\gamma \cdot \ell}$, where $\gamma > 0$ is a small constant, $t = O(1/\gamma)$ and huge field size $q = 2^{n/t}$ so that the input length of p (in bits) is $t \log q = n$ and the output length is $\log q = \Omega(n)$. It follows that p can be computed in time $2^{O(c\ell)} = n^{O(c)}$ and if we assume that E is hard for Σ_j-circuits then p cannot be computed by circuits of size $2^{\beta \cdot \ell} = n^{\beta c}$, which we can control by choosing c. The reader can also find this argument in the proof of Theorem 5.5 in [43].

From this, Lemma 5.1 in [43] gives that p (interpreted as a function with boolean input and output) is a function that is hard on average. This proves a version of Theorem 7.1 in which Σ_1 is replaced by Σ_2. The stronger statement (for Σ_1) follows because Lemma 5.1 in [43] also holds for $j = 1$. This was stated in an earlier version of [43] and follows by a more efficient implementation of the proof.

More specifically, the proof of Lemma 5.1 constructs the circuit C' by first specifying a randomized procedure which for every input $x \in \mathbb{F}^t$ computes $p(x)$ correctly with probability say $2/3$. The procedure requires "nonuniform advice" about p in the form of a point $z \in \mathbb{F}^t$ and its evaluation $p(z)$ (the existence of a "good point" z is shown in the proof). The

Goal: Construct a poly(n)-time computable $(n^{-b}, 2^{-\Omega(n)})$-rt-PRG, $G : \{0,1\}^n \to \{0,1\}^{n+1}$ for Σ_i-circuits of size n^b.

Assumption: E is hard for exponential size Σ_{i+3}-circuits.

Parameters:

- i, b, n - We are shooting to fool Σ_i-circuits of size n^b.
- We set $\rho = 2^{-\mu n}$ for a constant $0 < \mu < \frac{1}{2}$ to be specified later.
- We set $t = \nu n$ for a constant $\nu > \frac{1}{2}$ to be specified later.

Ingredients:

- We make use of an exponentially hard on average function due to Trevisan and Vadhan [43]. Specifically, let $b' = a \cdot b$ for a constant $a > 1$ to be chosen later. By Theorem 7.1, there exists a constant $\alpha > 0$ such that under the assumption that E is hard for exponential size Σ_{i+3}-circuits, for every sufficiently large n, there is a polynomial time computable function $f : \{0,1\}^t \to \{0,1\}^{t'=\alpha \cdot t}$ such that for every size $n^{b'}$, Σ_{i+2}-circuit A, $\Pr_{X \leftarrow U_t}[A(X) = f(X)] \leq 2^{-t'/3}$.
- A strong $(k = O(\log(1/\rho)), \rho/2000)$-extractor, $E : \{0,1\}^{t'} \times \{0,1\}^d \to \{0,1\}$, with seed length $d = O(\log(1/\rho))$. There are constructions of polynomial time computable extractors with these parameters [17]. (In fact, such extractors immediately follow from binary poly-time $(1/2 - \Omega(\rho), \text{poly}(1/\rho))$-list-decodable codes with rate poly(ρ) [42]. We have that $d = O(\log(1/\rho) = e \cdot \mu \cdot n$ for some constant $e > 1$. We now specify the constant ν (which was used to define $t = \nu \cdot n$) so that the equality $n = t + d$ holds, we will later choose $\mu > 0$ to be sufficiently small so that we indeed have that $\nu > 1/2$ as promised.

The rt-PRG: Define $g : \{0,1\}^t \times \{0,1\}^d \to \{0,1\}$, by $g(v, y) = E(f(v), y)$. The final generator $G : \{0,1\}^n \to \{0,1\}^{n+1}$ is given by $G(v, y) = (v, y, g(v, y))$.

Figure 2 An rt-PRG with one bit stretch.

computation of the procedure can be expressed in the following form: Given x, the procedure nondeterministically guesses polynomially many strings $h_1, \ldots, h_\ell$ of polynomial length. For each one it prepares a circuit T_i which depends on h_i (and also on $x, z, p(z)$). The procedure then uses approximate counting to verify that sufficiently many of the T_i's accept more than a fixed number of inputs.

Indeed, such a computation can be described (after removing the random coins) by a nondeterministic circuit that uses an NP-oracle to perform approximate counting. Overall, this gives a Σ_2-circuit.

However, approximate counting is only used to check that a given circuit accepts more than a fixed number of inputs. The problem of checking that the number of accepting inputs is larger than a fixed quantity is easier than approximating the number of accepting inputs: it can be solved by an Arthur-Merlin protocol as shown by Goldwasser and Sipser [16]. With this implementation, the entire procedure can be seen (after removing the randomness) as a Σ_1-circuit.

Our construction is given in Figure 2. Note that an intuitive overview is given in Section 2.

▶ **Theorem 7.2** (rt-PRG with one bit stretch)**.** *Let $i \geq 0, b > 1$ be constants. Assume that E is hard for exponential size Σ_{i+3} circuits. Let g, G be the functions constructed in Figure 2, with the parameters chosen there. Then there exists a constant $\mu > 0$ such that for every sufficiently large n, $G(x) = (x, g(x))$ is an (n^{-b}, ρ)-rt-PRG for size n^b, Σ_i-circuits, for $\rho = 2^{-\mu \cdot n}$. Furthermore, g is computable in time poly(n^b).*

7.1.2 Proof of Theorem 7.2

We start by proving the following lemma (which captures the role that strong seeded extractors play in the proof).

▶ **Lemma 7.3.** *Let* $E : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}$ *be a* (k,ρ)*-strong extractor, and let* $T : \{0,1\}^d \times \{0,1\} \to \{0,1\}$ *be a function. Let*

$$Z = \left\{ z : \Pr_{Y \leftarrow U_d}[T(Y, E(z,Y)) = 1] \overset{ad}{\not\sim}_{\rho} \Pr_{Y \leftarrow U_d, W \leftarrow U_1}[T(Y,W) = 1] \right\}.$$

Then $|Z| \leq 2 \cdot 2^k$.

Proof. We can partition Z into two sets according to which of the two terms in the definition of Z is the maximal one. If $|Z| > 2 \cdot 2^k$ then we can assume w.l.o.g

$$|\{z : \Pr_{Y \leftarrow U_d}[T(Y, E(z,Y)) = 1] - \Pr_{Y \leftarrow U_d, W \leftarrow U_1}[T(Y,W) = 1] > \rho\}| > 2^k.$$

Let X be a random variable that is uniformly distributed over Z and note that $H_\infty(X) > k$. E is a (k,ρ)-strong extractor which implies that $(Y, E(X,Y))$ is ρ-close to (Y,W). This implies that $|\Pr[T(Y, E(X,Y)) = 1] - \Pr[T(Y,W) = 1]| \leq \rho$, and we get a contradiction. ◀

We are now ready to prove Theorem 7.2. We assume (for contradiction) that G is not a (n^{-b},ρ)-rt-PRG for size n^b Σ_i-circuits, and our goal is to show that there is a Σ_{i+2}-circuit A of size $n^{b'}$ such that $\Pr_{X \leftarrow U_t}[A(X) = f(X)] > 2^{-t'/3}$.

Our assumption says that there exists a size n^b Σ_i-circuit C such that $\Pr[C(G(U_n)) = 1] \overset{rt}{\not\sim}_{(n^{-b},\rho)} \Pr[C(U_{n+1}) = 1]$. For the remainder of the proof, let us consider a probability space that consists of three independent random variables: $V \leftarrow U_t$, $Y \leftarrow U_d$ and $U \leftarrow U_1$. By the construction of G we have that:

$$\Pr[C(V, Y, E(f(V), Y)) = 1] \overset{rt}{\not\sim}_{(n^{-b},\rho)} \Pr[C(V,Y,U) = 1]$$

We now apply Lemma 4.10. For this purpose we set $R = V$, $W = (Y,U)$. We use the notation $W_1 = Y$ and $W_2 = U$. Let $\psi_1(r,w) = C(r, w_1, E(f(r), w_1))$ and $\psi_2(r,w) = C(r, w_1, w_2)$. We apply the lemma and obtain that:

▶ **Claim 7.4.** *Let* $\epsilon' = n^{-b}/10, \rho' = \rho \cdot n^{-b}/10$ *and let*

$$G = \left\{ v \in \{0,1\}^t : \Pr[C(v, Y, E(f(v), Y)) = 1] \overset{rt}{\not\sim}_{(\epsilon',\rho')} \Pr[C(v,Y,U) = 1] \right\}.$$

It follows that $\Pr[V \in G] \geq \rho \cdot n^{-b}/10$.

Recall that our goal is to contradict the hardness of f by showing the existence of a suitable circuit A that compute $f(v)$ too well given a random v. For every $v \in G$, we define $T_v(y,b) = C(v,y,b)$ so that T_v distinguishes $(Y, E(f(v),Y))$ from (Y,U) in the sense that $\Pr[T_v(Y, E(f(v),Y)) = 1] \overset{rt}{\not\sim}_{(\epsilon',\rho')} \Pr[T_v(Y,U) = 1]$. By Lemma 4.5, this implies that $\Pr[T_v(Y, E(f(v),Y)) = 1] \overset{ad}{\not\sim}_{\rho' \cdot \epsilon'/3} \Pr[T_v(Y,U) = 1]$. This is helpful because by Lemma 7.3 there aren't that many $z \in \{0,1\}^{t'}$ for which T_v distinguishes $(Y, E(z,Y))$ from (Y,U), and yet $z^* = f(v)$ is one of those z's. We will use this property to construct a small Σ_{i+2} circuit A that given a $v \in G$, produces $f(v)$ with probability $\rho^{O(1)}$, and this will be a contradiction. The description of A appears in Figure 3. We first present A as a randomized circuit that tosses coins, and will later fix its coins to give a circuit that does not toss coins. The correctness of A will follow from the following claims.

Goal: A size $n^{b'}$ Σ_{i+2}-circuit C that computes f with probability $2^{-t'/3}$.

Description: On input $v \in \{0,1\}^t$, A computes as follows:

- Prepare the Σ_i-circuit $T_v(y,b) = C(v,y,b)$
- Prepare the Σ_{i+1}-circuit B_v, defined as follows:
  - On input $z \in \{0,1\}^{t'}$, B_v computes as follows:
  - B_v prepares the circuits $D_1^v : \{0,1\}^d \to \{0,1\}$ and $D_2^v : \{0,1\}^{d+1} \to \{0,1\}$ defined by $D_1(y) = T_v(y, E(z,y))$ and $D_2(y,b) = T_v(y,b)$. Note that these are circuits of size $\text{poly}(n^b)$.
  - Let p_1, p_2 be the number of accepting inputs of D_1^v, D_2^v respectively. Let $\eta = \epsilon'/10 = n^{-b}/100$. B uses Theorem 4.2 to compute η-relative approximations p_1', p_2' to p_1, p_2. Note that by Theorem 4.2, this can be done in a size $\text{poly}(n^b)$ Σ_{i+1}-circuit.
  - B_v accepts z if $\max(p_1', p_2') \geq \rho' e^{-\eta}$ and $\max(p_1', p_2')/\min(p_1', p_2') \geq e^{\epsilon' - 2\eta}$. This choice is made so that by Lemma 4.9:
    - $p_1 \overset{rt}{\not\sim}_{(\epsilon',\rho')} p_2 \Rightarrow B_v$ accepts z, and
    - B_v accepts $z \Rightarrow p_1 \overset{rt}{\not\sim}_{(\epsilon'',\rho'')} p_2$, where $\epsilon'' = \epsilon' - 4\eta \geq \epsilon'/2$ and $\rho'' = \rho' \cdot e^{-2\eta} \geq \rho'/2$.
- The circuit A uses Theorem 4.3 to sample an accepting input z of B_v (with error $\delta = 2^{-n}$). Note that this can be done with a size $\text{poly}(n^b)$ Σ_{i+2}-circuit. Overall, the size of A is $\text{poly}(n^b)$. Recall that in Figure 2 we have chosen $b' = a \cdot b$ for an unspecified constant a. Note that A is indeed a Σ_{i+2}-circuit, and we can now choose a to be sufficiently large so that the size of A is $\text{poly}(n^b) = n^{a \cdot b} = n^{b'}$.
- Finally, the circuit A outputs z.

Figure 3 Circuit A: A Σ_{i+2}-circuit that computes f correctly with noticeable probability.

▶ **Claim 7.5.** *For every $v \in G$, B_v accepts $f(v)$.*

Proof. Fix some $v \in G$. By Claim 7.4 and the definition of T_v, $\Pr[T_v(Y, E(f(v), Y)) = 1] \overset{rt}{\not\sim}_{(\epsilon',\rho')} \Pr[T_v(Y,U) = 1]$. When B_v receives $z = f(v)$ as input, it prepares the circuit $D_1(y) = T_v(y, E(f(v), y))$ and $D_2(y,b) = T_v(y,b)$. Recall that p_1, p_2 are the number of accepting inputs of these two circuits. Therefore, we have that $p_1 \overset{rt}{\not\sim}_{(\epsilon',\rho')} p_2$, and by construction, B_v accepts z. ◀

▶ **Claim 7.6.** *For every $v \in G$, B_v accepts at most $2^{k+1} = (\frac{1}{\rho})^{O(1)}$ inputs.*

Proof. By construction, if B_v accepts an input z, then the quantities p_1, p_2 in Figure 3, satisfy $p_1 \overset{rt}{\not\sim}_{(\epsilon' - 4\eta, \rho' \cdot e^{-2\eta})} p_2$. By Lemma 4.5, this implies that $p_1 \overset{ad}{\not\sim}_{\epsilon'' \cdot \rho''/3} p_2$, and note that

$$\epsilon'' \cdot \rho''/3 \geq \epsilon' \cdot \rho'/12 \geq n^{-2b} \cdot \rho/1200 \geq \rho/2000.$$

Therefore, if B_v accepts z, then $\Pr[T_v(Y, E(z,Y)) = 1] \overset{ad}{\not\sim}_{\rho/2000} \Pr[T_v(Y,U) = 1]$ and by Lemma 7.3, there are at most $2 \cdot 2^k = \text{poly}(1/\rho)$ such strings z. ◀

The two claims above give that

▶ **Claim 7.7.** *For every $v \in G$, $\Pr[A(v) = f(v)] \geq 2^{-(k+1)} = \rho^{O(1)}$ (where the probability is over the coin tosses of A).*

It follows that

$$\Pr[A(V) = f(V)] \geq \Pr[V \in G] \cdot \Pr[A(V) = f(V) | V \in G] \geq$$
$$\rho \cdot n^{-b}/10 \cdot \rho^{O(1)} = \rho^{O(1)} = 2^{-O(\mu \cdot n)} \geq 2^{-t'/3},$$

where the last step follows because we can choose $\mu > 0$ to be sufficiently small so that

$$t'/3 = \alpha \cdot t/3 = \alpha \cdot \nu \cdot n/3 \geq \alpha \cdot n/6 \geq O(\mu n).$$

Finally, by an averaging argument, we can hardwire the random coins of A to produce a non-randomized circuit with the same success probability. Thus, the circuit A contradicts the assumption that f is $2^{-t'/3}$-incomputable by Σ_{i+2}-circuits of size $n^{b'}$.

7.2 rt-PRGs with polynomial stretch

We now use the NW-generator to transform the 1-bit stretch rt-PRG into an rt-PRG with polynomial stretch.

7.2.1 Using the NW-generator

We review the construction of the NW-generator.

▶ **Definition 7.8** (Design). A collection $\Delta = (S_1, \ldots, S_n)$ of subsets of $[r]$ is an (r, ℓ, u)-design if

- For every $i \in [n]$, $|S_i| = \ell$.
- For every distinct $i, j \in [n]$, $|S_i \cap S_j| \leq u$.

▶ **Definition 7.9** (NW-generator). Let $\Delta = (S_1, \ldots, S_n)$ be an (r, ℓ, u)-design, and let $g : \{0,1\}^\ell \to \{0,1\}$ be a function. For $y \in \{0,1\}^r$, we define $x_i(y) = y|_{S_i}$ and $z_i(y) = g(x_i(y))$. Let,

$$\mathrm{NW}_g^\Delta(y) = z_1(y), \ldots, z_n(y).$$

Theorem 1.17 follows from the next theorem.

▶ **Theorem 7.10.** *Let Δ be an (r, ℓ, u)-design with $u = c \cdot \log n$. If $G(x) = (x, g(x))$ is an $(\frac{\epsilon}{20n}, \frac{\rho \cdot \epsilon}{30n})$-rt-PRG for size $n^{c+1} + n^b + O(n)$ circuits, then $NW_g^\Delta : \{0,1\}^r \to \{0,1\}^n$ is an (ϵ, ρ)-rt-PRG for size n^b circuits.*

We need the following notation for the proof.

▶ **Definition 7.11.** Given $x \in \{0,1\}^\ell$, $v \in \{0,1\}^{r-\ell}$ and $i \in [n]$ let $y^{(i)}(x, v)$ denote the r-bit string y obtained by "placing" the bits of x in the ℓ indices of y that are in S_i, and using v to fill the remaining $r - \ell$ positions.

of Theorem 7.10. In this proof g and Δ are fixed, and so, to avoid clutter we write NW instead of NW_g^Δ. Assume for contradiction that NW is not an (ϵ, ρ)-rt-PRG for size n^b circuits. That is, that there exists a circuit D of size n^b such that

$$\Pr[D(\mathrm{NW}(U_r)) = 1] \overset{rt}{\not\sim}_{(\epsilon,\rho)} \Pr[D(U_n) = 1].$$

▶ **Claim 7.12** ("Relative error hybrid argument"). *Consider a probability space consisting of independent random variables $Y \leftarrow U_r$ and $B_1, \ldots, B_n \leftarrow U_1$, and define*

$$H_i = z_1(Y), \ldots, z_i(Y), B_{i+1}, \ldots, B_n,$$

so that $H_0 = U_n$ and $H_n = NW(Y)$. There exists $0 \leq i < n$ such that

$$\Pr[D(H_i) = 1] \overset{rt}{\not\sim}_{(\epsilon/2n, \rho \cdot e^{-\epsilon}/2)} Pr[D(H_{i+1}) = 1]$$

Proof. Let $p_i = \Pr[D(H_i) = 1]$. We have that $p_0 \overset{rt}{\not\sim}_{(\epsilon,\rho)} p_n$ which indeed implies that there exists an i such that $p_i \overset{rt}{\not\sim}_{(\epsilon/2n,\rho\cdot e^{-\epsilon}/2)} p_{i+1}$. ◀

We have that

$$\Pr[D(z_1(Y),\ldots,z_i(Y),B_{i+1},\ldots,B_n)=1] \overset{rt}{\not\sim}_{(\epsilon/2n,\rho\cdot e^{-\epsilon}/2)}$$
$$\Pr[D(z_1(Y),\ldots,z_{i+1}(Y),B_{i+2},\ldots,B_n)=1$$

We can imagine that the experiment of choosing $Y \leftarrow U_r$ is performed by choosing independently $X \leftarrow U_\ell$ and $V \leftarrow U_r$ and setting $Y = y^{(i+1)}(X,V)$. We now apply Lemma 4.10 setting $R = (V, B_{i+2},\ldots,B_n)$ and $W = (X, B_{i+1})$. We conclude that there exists a fixing $(v, b_{i+2},\ldots,b_n)$ for R such that

$$\Pr[D(z_1(y^{(i+1)}(X,v)),\ldots,z_i(y^{(i+1)}(X,v)),B_{i+1},b_{i+2},\ldots,b_n)=1] \overset{rt}{\not\sim}_{(\epsilon/20n,\rho\cdot e^{-\epsilon}\cdot\epsilon/20n)}$$

$$\Pr[D(z_1(y^{(i+1)}(X,v)),\ldots,z_{i+1}(y^{(i+1)}(X,v)),b_{i+2},\ldots,b_n)=1].$$

Note that by definition, $z_{i+1}(y^{(i+1)}(X,v)) = g(X)$. Furthermore, note that as Δ is a (r,ℓ,u)-design, for $j \le i$, $z_j(y^{(i+1)}(X,v)$ depends only on u bits of X, and therefore, can be computed by a circuit $C_j(X)$ of size 2^u. We now define a circuit C as follows:

$$C(x,b) = D(C_1(x),\ldots,C_i(x),b,b_{i+2},\ldots,b_n).$$

Substituting in the expression above, we have that:

$$\Pr[C(X,g(X))=1] \overset{rt}{\not\sim}_{(\epsilon/20n,\rho\cdot e^{-\epsilon}\cdot\epsilon/20n)} \Pr[C(X,B_{i+1})=1]$$

Note that C is of size $n\cdot 2^u + n^b + O(n) = n^{c+1} + n^b + O(n)$ and that $\rho\cdot e^{-\epsilon}\cdot\epsilon/20n \ge \rho\cdot\epsilon/30n$. ◀

7.2.2 Putting it together: Proof of Theorem 1.17

We assume that E is hard for exponential size Σ_{i+3}-circuits. Let $e, b > 1$ be some constants. Nisan and Wigderson [29] showed that there exists a constant $c > 1$ such that for every sufficiently large n, there is an (r,ℓ,u) design with $n = r^e$ sets, that has $r = O(\ell^2)$ and $u = c\log n$. Note that $n = O(\ell^{2e})$ and so, by Theorem 7.2 there are polynomial time computable functions $g : \{0,1\}^\ell \to \{0,1\}$ and $G(x) = (x, g(x))$ such that G is a $(n^{-b'},\rho)$-rt-PRG for size $n^{b'}$ Σ_i-circuits, where b' is a constant that we choose later, and $\rho = 2^{-\Omega(\ell)} = 2^{-\Omega(\sqrt{r})}$. We can choose the constant b' to be sufficiently large so that by Theorem 7.10 we have that NW_g^Δ is an (ϵ',ρ')-rt-PRG for size n^b Σ_i-circuits, with $\epsilon' = n^{-b'}\cdot 20n \le n^{-b}$ and $\rho' = \rho\cdot 30n/n^{-b'} = \rho^{\Omega(1)} = 2^{-\Omega(\sqrt{r})}$ for sufficiently large n. This gives a re-PRG with the same parameters.

8 A construction of re-nb-PRGs

In this section we construct rt-nb-PRGs which imply re-nb-PRGs.

8.1 rt-PRGs for Σ_1-circuits are rt-nb-PRGs

We first show that sufficiently strong rt-PRGs for Σ_1-circuits are rt-nb-PRGs. More specifically that (ϵ, ρ)-rt-PRGs with $\rho = O(2^{-\ell} \cdot \rho' \cdot \epsilon)$ for Σ_1-circuits are $(\ell, O(\epsilon), \rho')$-rt-nb-PRGs (for standard circuits).

▶ **Lemma 8.1.** *There exists a constant $c > 1$ such that for every constant $b > 1$, and for every sufficiently large n, if $G : \{0,1\}^r \to \{0,1\}^n$ is an (ϵ, ρ)-rt-PRG for Σ_1-circuits of size n^{bc} with $\rho \leq 2^{-(\ell+t)}$ then G is a $(\ell, 4 \cdot \epsilon, \rho')$-rt-nb-PRG for circuits of size n^b for $\rho' = O(2^{-t}/\epsilon)$.*

Proof. Let $C : \{0,1\}^n \to \{0,1\}^\ell$ be a size n^b circuit. We consider the circuit $A : \{0,1\}^\ell \to \{0,1\}$ that on input $z \in \{0,1\}^\ell$ computes a 1/10-relative approximation to the quantity $\Pr[C(U_n) = z]$ and accepts if and only if the approximation is smaller than 2ρ. By Theorem 4.2, A can be implemented by a Σ_1-circuit of size $\text{poly}(n^b)$. We use A also to denote the set of inputs accepted by A. Note that for every $z \in A$, $\Pr[C(U_n) = z] < 4\rho$. It follows that $\Pr[C(U_n) \in A] \leq 2^\ell \cdot 4\rho = 2^{-(t-2)}$. By the pseudorandomness of G, this implies that $\Pr[C(G(U_r)) \in A] \leq e^\epsilon 2^{-(t-2)}$.

Let $H = \{z : \Pr[C(U_n) = z] \geq 4\rho\}$. We have that $H \cap A = \emptyset$. For every $z \in \{0,1\}^\ell$, we can consider the circuit $T_z(x)$ which accepts iff $C(x) = z$. This circuit is fooled by G. For $z \notin A$, $\Pr[C(U_n) = z] \geq \rho$, and we have that $\Pr[C(U_n) = z] \overset{rt}{\sim}_{(\epsilon,0)} \Pr[C(G(U_r)) = z]$. This in turn implies that for every T such that $T \cap A = \emptyset$, $\Pr[C(U_n) \in T] \overset{rt}{\sim}_{(\epsilon,0)} \Pr[C(G(U_r)) \in T]$. We will show that:

▶ **Claim 8.2.** *For every $D \subseteq \{0,1\}^\ell$, $\Pr[C(U_n) \in D] \overset{re}{\sim}_{(\epsilon,\delta)} \Pr[C(G(U_r)) \in D]$ for $\delta = 2^{-(t-2)}$.*

Proof. We have that:

$$\begin{aligned}\Pr[C(U_n) \in D] &= \Pr[C(U_n) \in D \setminus H] + \Pr[C(U_n) \in D \cap H] \\ &\leq 4\rho \cdot 2^\ell + e^\epsilon \cdot \Pr[C(G(U_r)) \in D \cap H] \\ &\leq 2^{-(t-2)} + e^\epsilon \cdot \Pr[C(G(U_r)) \in D \cap H]\end{aligned}$$

We also have that:

$$\begin{aligned}\Pr[C(U_n) \in D] &\geq \Pr[C(U_n) \in D \setminus A] \\ &\geq e^{-\epsilon} \cdot \Pr[C(G(U_r)) \in D \setminus A] \\ &= e^{-\epsilon} \cdot (\Pr[C(G(U_r)) \in D] - \Pr[C(G(U_r)) \in D \cap A]) \\ &\geq e^{-\epsilon} \cdot (\Pr[C(G(U_r)) \in D] - e^\epsilon \cdot 2^{-(t-2)}) \\ &= e^{-\epsilon} \cdot \Pr[C(G(U_r)) \in D] - 2^{-(t-2)}\end{aligned}$$

◀

The lemma now follows because by Lemma 4.5 for $\epsilon \leq \frac{1}{2}$, $p_1 \overset{re}{\sim}_{(\epsilon,\delta)} p_2 \Rightarrow p_1 \overset{rt}{\sim}_{(4\epsilon, 4\delta/\epsilon)} p_2$.

◀

Using the rt-PRG of Theorem 1.17 we obtain the following rt-nb-PRG.

▶ **Theorem 8.3.** *Let $b, e > 1$ be constants, and $\ell = \ell(n) \leq n$, $\rho = \rho(n)$ be functions. If E is hard for exponential size Σ_4-circuits, then there is a polynomial time computable $G : \{0,1\}^r \to \{0,1\}^n$, such that for every sufficiently large n, G is an (ℓ, n^{-b}, ρ)-rt-nb-PRG for circuits of size n^b, with $r = O((\ell + \log(1/\rho))^2)$.*

This is disappointing as previous work on (non-relative) nb-PRGs achieves a better dependence on ℓ in the form of $r = O(\ell)$. We would like to also achieve a linear dependence of r on ℓ.

8.2 An rt-nb-PRG with seed length $r = \ell + O(\log(1/\rho))^2$

We will use the approach of Theorem 1.18 to achieve a construction with shorter seed length. Specifically, we design a poly(n) time randomized procedure P that produces circuit that is with high probability an rt-PRG for Σ_1-circuits that has excellent seed length. We then show that checking whether a given circuit is an rt-PRG for Σ_1-circuits can be done in the polynomial time hierarchy. This means that our rt-PRG G' of the earlier section can be used to produce a circuit h that with high probability is an rt-PRG for Σ_1-circuits. This in turn implies that it is an rt-nb-PRG for standard circuits. Our final rt-PRGs takes two seeds, x_1, x_2. It first constructs h by applying $P(G'(x_1))$ and then outputs $h(x_2)$.

8.2.1 A random hash function is an rt-PRG

We use the following standard construction of t-wise independent hash functions (that is based on degree $t-1$ polynomials).

▶ **Theorem 8.4** (t-wise independent hash functions)**.** *For every n, m, t there is a family $\mathcal{H}^t_{n,m}$ of at most $2^{d=t\cdot\max(n,m)}$ functions from n bits to m bits, such that for every distinct $x_1, \ldots, x_t \in \{0,1\}^n$, the random variables $h(x_1), \ldots, h(x_t)$ defined over the experiment $h \leftarrow \mathcal{H}^t_{n,m}$ are uniformly distributed and t-wise independent. Furthermore, there is a polynomial time algorithm that given the d bit description s of a hash function $h_s \in \mathcal{H}^t_{n,m}$, and an input $x \in \{0,1\}^n$, computes $h_s(x)$.*

A standard probabilistic argument shows that for any class $\mathcal{C}$ with 2^k functions, a random function $G : \{0,1\}^r \to \{0,1\}^n$ is w.h.p. a PRG for $\mathcal{C}$ with $r \approx \log k$. In the theorem below, we repeat this argument and show that it also applies for rt-PRGs and achieves an excellent dependence on ρ.

▶ **Theorem 8.5.** *Let $\mathcal{C}$ be a family of at most 2^k boolean functions on n bits. Let $t = 2(k+3) + 2n$ and $r = \log k + \log n + 2\log(1/\epsilon) + \log(1/\rho) + c$ for a sufficiently large universal constant c. With probability at least $1 - 2^{-n}$ over $h \leftarrow \mathcal{H}^t_{r,n}$, we obtain a function $h : \{0,1\}^r \to \{0,1\}^n$ that is an (ϵ, ρ)-rt-PRG for $\mathcal{C}$.*

Proof. Let $C : \{0,1\}^n \to \{0,1\}$ be a function in $\mathcal{C}$, and let $\mu = \Pr[C(U_n) = 1]$. Let X_C be the random variable that counts the number of $s \in \{0,1\}^r$ such that $C(h(s)) = 1$. The random variable X_C is a sum of 2^r, t-wise independent variables. We have that $E(X_C) = 2^r \cdot \mu$. By the t-wise independent "Chernoff style" bound of Bellare and Rompel [8] we have that for even t,

$$\Pr[|X_C - E(X_C)| \geq A] \leq 8 \cdot \left(\frac{t \cdot E(X_C) + t^2}{A^2}\right)^{t/2}$$

Let $A = \frac{1}{3} \cdot \epsilon \cdot 2^r \cdot \max(\mu, \rho)$. This choice is made so that $\frac{X_C}{2^r} \overset{rt}{\not\sim}_{(\epsilon,\rho)} \mu$ implies that $|X_C - E(X_C)| \geq A$. By our choice of parameters it follows that:

$$\frac{t \cdot E(X_C) + t^2}{A^2} \leq \frac{t \cdot \mu \cdot 2^r + t^2}{\frac{1}{9} \cdot \epsilon^2 \cdot 2^{2r} \cdot \max(\mu, \rho)^2} \leq \frac{t}{\frac{1}{9} \cdot \epsilon^2 \cdot 2^r \cdot \rho} + \left(\frac{t}{\frac{1}{3} \cdot \epsilon \cdot 2^r \cdot \rho}\right)^2 \leq \frac{1}{2}$$

where the last inequality follows for a sufficiently large constant c in the definition of r, and

using our choice of parameters. It follows that

$$\begin{aligned}\Pr[\frac{X_C}{2^r} \overset{rt}{\not\sim}_{(\epsilon,\rho)} \mu] &\leq 8 \cdot \left(\frac{t \cdot E(X_C) + t^2}{A^2}\right)^{t/2} \\ &\leq 8 \cdot \left(\frac{1}{2}\right)^{t/2} \\ &\leq 2^{-n} \cdot 2^{-k}\end{aligned}$$

where the last inequality follows by our choice of t. By a union bound over all 2^k functions C in $\mathcal{C}$ we have that with probability $1 - 2^{-n}$ we obtain an rt-PRG. ◀

▶ **Corollary 8.6.** *For every $b > 1$, and for every sufficiently large n, and every $\epsilon, \rho \geq 2^{-n}$, there is a randomized Turing Machine P running in time poly(n^b) that with probability $1-2^{-n}$ produces a circuit $h : \{0,1\}^r \to \{0,1\}^n$ that is an (ϵ, ρ)-rt-PRG for Σ_1-circuits of size n^b, with $r = O(b \log n) + 2\log(1/\epsilon) + \log(1/\rho)$.*

8.2.2 The complexity of checking if a given circuit is an rt-PRG

We consider the problem of checking whether a given circuit is an rt-PRG. We would like to show that this problem is in the polynomial time hierarchy. The following formulation as a promise problem makes this possible, and will suffice for our needs.

▶ **Definition 8.7.** Let $\mathsf{DC}^{i,s,s'}_{\epsilon,\rho}$ denote the following promise problem:
Input: a circuit $G : \{0,1\}^r \to \{0,1\}^n$ of size s.
Yes instances: G is not an (ϵ, ρ)-rt-PRG for Σ_i-circuits of size s'.
No instances: G is an $(\epsilon/2, \rho \cdot (1-\epsilon))$-rt-PRG for Σ_i-circuits of size s'.

▶ **Theorem 8.8.** *For every $i \geq 0$, $0 < \epsilon, \rho \leq 1$, and $r \leq n \leq s' \leq s$ there is a nondeterministic Σ_{i+2}-circuit of size poly$(r, n, s, 1/\epsilon)$ which solves $\mathsf{DC}^{i,s,s'}_{\epsilon,\rho}$.*

Proof. We consider the following nondeterministic Σ_{i+1}-circuit A: when given the circuit G as input, the circuit A guesses a Σ_i-circuit $C : \{0,1\}^n \to \{0,1\}$ of size s'. Let $\eta = \epsilon/10$. Using Theorem 4.2, A computes η-relative approximations p'_1, p'_2 of the quantities of $p_1 = \Pr[C(U_n) = 1]$ and $p_2 = \Pr[C(G(U_n) = 1]$. A then applies the test $T(p'_1, p'_2)$ of Lemma 4.9 and outputs its outcome. By Lemma 4.9:

- $p_1 \overset{rt}{\not\sim}_{(\epsilon,\rho)} p_2 \Rightarrow T(p'_1, p'_2)$ accepts.
- $T(p'_1, p'_2)$ accepts $\Rightarrow p_1 \overset{rt}{\not\sim}_{(\epsilon-4\eta,\rho\cdot e^{-2\eta})} p_2$.

The theorem follows by our choice of η. ◀

The key is that the size of the circuit above does not depend on ρ, and note that if the circuit rejects G, then G is an (ϵ, ρ)-rt-PRG.

8.3 rt-nb-PRGs with small seed length

The following Theorem implies Theorem 1.21.

▶ **Theorem 8.9** (rt-nb-PRG with seed length $1 \cdot \ell + O(\log(1/\rho))^2$)**.** *Let $b > 1$ and $\alpha > 0$ be constants and $\ell = \ell(n) \leq n$, $\rho = \rho(n) \leq 2^{-n^\alpha}$. Assume that E is hard for exponential size Σ_6-circuits. Let G be the function constructed in Figure 4, with the parameters chosen there. Then for every sufficiently large n, there is a polynomial time computable (ℓ, n^{-b}, ρ)-rt-nb-PRG $G : \{0,1\}^r \to \{0,1\}^n$ for size n^b circuits, with $r = \ell + O(\log(1/\rho))^2$.*

Goal: Construct a poly(n)-time computable (ℓ, n^{-b}, ρ)-rt-nb-PRG, $G : \{0,1\}^r \to \{0,1\}^n$ for circuits $C : \{0,1\}^n \to \{0,1\}^\ell$ of size n^b.

Assumption: E is hard for exponential size Σ_6-circuits.

Parameters:

- ℓ, b, n - We are shooting to fool circuits $C : \{0,1\}^n \to \{0,1\}^\ell$ of size n^b.
- We require that $\rho \leq 2^{-n^\alpha}$ for some constant $\alpha > 0$. (This is done to simplify the presentation).

Ingredients:

- We make use of the Turing Machine P of Corollary 8.6. Specifically, let $b' = a \cdot b$ for a sufficiently large constant $a > 1$ to be chosen later. By Corollary 8.6 there is a randomized Turing Machine P running in time poly$(n^{b'})$ which produces a circuit $h : \{0,1\}^r \to \{0,1\}^n$, that with probability $1 - 2^{-n}$ is an $(n^{-b'}, \rho' = 2^{-\ell} \cdot \rho \cdot n^{-3b'})$-rt-PRG for Σ_1-circuits of size $n^{b'}$, and $r_1 = \ell + O(b' \cdot \log n) + \log(1/\rho)$.
- We also make use of the rt-PRG of Theorem 1.17. Specifically, let $b'' = a \cdot b'$ and recall that a sufficiently large constant $a > 1$ will be chosen later. By Theorem 1.17 the hardness assumption that E is hard for exponential size Σ_6-circuits implies that there is a poly$(n^{b''})$ computable $(\frac{1}{2}, \rho')$-rt-PRG $G' : \{0,1\}^{r_2} \to \{0,1\}^{n^{b''}}$ for size $n^{b''}$ Σ_3-circuits, with $r_2 = O(1/\rho))^2$. (Here we use the fact that $\rho \leq 2^{-n^\alpha}$ so that $\log(1/\rho) \geq n^\alpha$).

The rt-nb-PRG:

- Let $r = r_1 + r_2 = \ell + O(b' \cdot \log n) + O(\log(1/\rho))^2 = \ell + O(\log(1/\rho))^2$. Given $x \in \{0,1\}^r$ interpret it as $(x_1, x_2) \in \{0,1\}^{r_1} \times \{0,1\}^{r_2}$.
- Run the procedure P using the string $G'(x_2)$ as random coins. (Note that we can choose the constant a to be sufficiently large so that $n^{b''} = n^{ab'}$ is larger than the number of coins required by P). The procedure P produces a circuit $h : \{0,1\}^{r_1} \to \{0,1\}^n$.
- $G(x)$ outputs $h(x_1)$.

Figure 4 An rt-nb-PRG with seed length $\approx 1 \cdot \ell$.

Proof of Theorem 8.9. We first argue, that when G applies P to obtain a circuit h, then w.h.p. it obtains an rt-PRG. Specifically,

▶ **Claim 8.10.** *With probability $1 - 2\rho'$ over $x_2 \leftarrow U_{r_2}$, the circuit $h : \{0,1\}^{r_1} \to \{0,1\}^n$ obtained by $P(G'(x_2))$ is a $(n^{-b'}, \rho')$-rt-PRG for size $n^{b'}$ Σ_1-circuits.*

Proof. (of claim) Let $s' = n^{b'}$ and $s = \text{poly}(n^{b'})$ be a bound on the size of h. By Theorem 8.3 we have that the promise problem $\mathsf{DC}^{1,s,s'}_{n^{-2b'},\rho'}$ is solved by a nondeterministic Σ_3-circuit T of size poly$(n^{b'})$. Recall that if T rejects a given circuit, then this circuit is a $(n^{-2b'}, \rho')$-rt-PRG for Σ_1-circuits of size $s' = n^{b'}$. Let $D(z) = T(P(z))$ and note that D can be implemented by a Σ_3-circuit of size poly$(n^{b'})$. The parameters of the generator G' were chosen so that it fools D. More specifically, by choosing a to be sufficiently large we have that the size of D is smaller than $n^{b''} = n^{a \cdot b'}$. By the guarantee on P, we know that the probability that D accepts a uniform input is at most 2^{-n}. As G' is a $(\frac{1}{2}, \rho')$-PRG it follows that the probability that D accepts $G'(U_{r_2})$ is at most $e^{\frac{1}{2}} \cdot \rho' \leq 2\rho'$. The claim follows. ◀

We have that with probability $1 - 2\rho'$ over the choice of x_2, G output $h(U_{r_1})$ for h that is a $(n^{-b'}, \rho')$-rt-PRG for Σ_1-circuits of size $n^{b'}$. This implies that G is a $(O(n^{-b'}), O(\rho'/n^{-b'}))$-rt-PRG for size $n^{b'}$ Σ_1-circuits. A trivial (albeit somewhat wasteful) way to see this is to use Lemma 4.5 to transform the guarantee on $\overset{rt}{\sim}_{(,)}$ to $\overset{re}{\sim}_{(,)}$ which makes the calculation straightforward, and then transform back. This is why we have $n^{-b'}$ in the denominator.

Finally, we choose a to be sufficiently large so that by Lemma 8.1 an rt-PRG against Σ_1-circuits of size $n^{b'} = n^{ab}$ is a rt-nb-PRG for circuits of size n^b. Applying the lemma, we get that G is an $(O(n^{-b'}), O(\rho'/n^{-2b'}))$-rt-nb-PRG for circuits of size n^b. The Theorem follows as by our choice of parameters $O(n^{-b'})$ can be made smaller than n^{-b} and $O(\rho'/n^{-2b'})$ is smaller than ρ. ◀

9 Open Problems

Theorem 1.8 is the first hardness versus randomness tradeoff that is applicable to randomized algorithm solving NP complete problem. It is interesting to find more instances where this approach can be used to efficiently derandomize algorithms for other NP complete problems. Is it possible to give hardness versus randomness tradeoffs for general randomized algorithms (that are not necessarily OPP)?

The dependence of the seed length on the parameter δ in Theorem 1.17 is additive in $\log(1/\delta)^2$ can this be reduced to $\log(1/\delta)$? As explained in the introduction, this will give improvements in applications of the theorem.

Can we find more applications of re-PRGS and re-nb-PRGs? It will be especially interesting to find cryptographic applications in computational settings as discussed in Section 1.14.

Acknowledgements. This work was done in part while the last three authors were visiting Simons Institute for the Theory of Computing. We are grateful to anonymous referees for helpful comments.

References

1 Noga Alon, Raphael Yuster, and Uri Zwick. Color-coding. *J. ACM*, 42(4):844–856, 1995.
2 Benny Applebaum, Sergei Artemenko, Ronen Shaltiel, and Guang Yang. Incompressible functions, relative-error extractors, and the power of nondeterministic reductions (extended abstract). In *Conference on Computational Complexity*, volume 33 of *LIPIcs*, pages 582–600. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2015.
3 Sergei Artemenko and Ronen Shaltiel. Lower bounds on the query complexity of non-uniform and adaptive reductions showing hardness amplification. *Computational Complexity*, 23(1):43–83, 2014.
4 Sergei Artemenko and Ronen Shaltiel. Pseudorandom generators with optimal seed length for non-boolean poly-size circuits. In *STOC*, pages 99–108. ACM, 2014.
5 László Babai, Lance Fortnow, Noam Nisan, and Avi Wigderson. BPP has subexponential time simulations unless EXPTIME has publishable proofs. *Computational Complexity*, 3:307–318, 1993.
6 Boaz Barak, Shien Jin Ong, and Salil P. Vadhan. Derandomization in cryptography. *SIAM J. Comput.*, 37(2):380–400, 2007.
7 Mihir Bellare, Oded Goldreich, and Erez Petrank. Uniform generation of NP-witnesses using an NP-oracle. *Inf. Comput.*, 163(2):510–526, 2000.
8 Mihir Bellare and John Rompel. Randomness-efficient oblivious sampling. In *FOCS*, pages 276–287. IEEE Computer Society, 1994.
9 Manuel Blum and Silvio Micali. How to generate cryptographically strong sequences of pseudo-random bits. *SIAM J. Comput.*, 13(4):850–864, 1984.
10 Andrew Drucker. Nondeterministic direct product reductions and the success probability of SAT solvers. In *FOCS*, pages 736–745. IEEE Computer Society, 2013.

11 Bella Dubrov and Yuval Ishai. On the randomness complexity of efficient sampling. In *STOC*, pages 711–720. ACM, 2006.

12 Uriel Feige and Carsten Lund. On the hardness of computing the permanent of random matrices. *Computational Complexity*, 6(2):101–132, 1997.

13 Oded Goldreich. A sample of samplers: A computational perspective on sampling. In *Studies in Complexity and Cryptography*, volume 6650 of *Lecture Notes in Computer Science*, pages 302–332. Springer, 2011.

14 Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In *STOC*, pages 25–32. ACM, 1989.

15 Oded Goldreich and Avi Wigderson. Derandomization that is rarely wrong from short advice that is typically good. In *RANDOM*, volume 2483 of *Lecture Notes in Computer Science*, pages 209–223. Springer, 2002.

16 Shafi Goldwasser and Michael Sipser. Private coins versus public coins in interactive proof systems. In *STOC*, pages 59–68. ACM, 1986.

17 Venkatesan Guruswami, Christopher Umans, and Salil P. Vadhan. Unbalanced expanders and randomness extractors from parvaresh–vardy codes. *J. ACM*, 56(4), 2009.

18 Dan Gutfreund and Guy N. Rothblum. The complexity of local list decoding. In *APPROX-RANDOM*, volume 5171 of *Lecture Notes in Computer Science*, pages 455–468. Springer, 2008.

19 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.

20 Yenjo Han, Lane A. Hemaspaandra, and Thomas Thierauf. Threshold computation and cryptographic security. *SIAM J. Comput.*, 26(1):59–78, 1997.

21 Russell Impagliazzo. Hard-core distributions for somewhat hard problems. In *FOCS*, pages 538–545. IEEE Computer Society, 1995.

22 Russell Impagliazzo, William Matthews, and Ramamohan Paturi. A satisfiability algorithm for ac^0. In *SODA*, pages 961–972. SIAM, 2012.

23 Russell Impagliazzo, Ronen Shaltiel, and Avi Wigderson. Near-optimal conversion of hardness into pseudo-randomness. In *FOCS*, pages 181–190. IEEE Computer Society, 1999.

24 Russell Impagliazzo, Ronen Shaltiel, and Avi Wigderson. Reducing the seed length in the nisan-wigderson generator. *Combinatorica*, 26(6):647–681, 2006.

25 Russell Impagliazzo and Avi Wigderson. $P = BPP$ if E requires exponential circuits: Derandomizing the XOR lemma. In *STOC*, pages 220–229. ACM, 1997.

26 Mark Jerrum, Leslie G. Valiant, and Vijay V. Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theor. Comput. Sci.*, 43:169–188, 1986.

27 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.

28 Peter Bro Miltersen and N. V. Vinodchandran. Derandomizing arthur-merlin games using hitting sets. *Computational Complexity*, 14(3):256–279, 2005.

29 Noam Nisan and Avi Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994.

30 Ramamohan Paturi and Pavel Pudlák. On the complexity of circuit satisfiability. In *STOC*, pages 241–250. ACM, 2010.

31 Ramamohan Paturi, Pavel Pudlák, Michael E. Saks, and Francis Zane. An improved exponential-time algorithm for k-SAT. *J. ACM*, 52(3):337–364, 2005.

32 Ramamohan Paturi, Pavel Pudlák, and Francis Zane. Satisfiability coding lemma. *Chicago J. Theor. Comput. Sci.*, 1999, 1999.

33 Rahul Santhanam. Fighting perebor: New and improved algorithms for formula and QBF satisfiability. In *FOCS*, pages 183–192. IEEE Computer Society, 2010.

34 Uwe Schöning. A probabilistic algorithm for k-SAT and constraint satisfaction problems. In *FOCS*, pages 410–414. IEEE Computer Society, 1999.

35 Ronen Shaltiel and Christopher Umans. Simple extractors for all min-entropies and a new pseudorandom generator. *J. ACM*, 52(2):172–216, 2005.

36 Ronen Shaltiel and Christopher Umans. Pseudorandomness for approximate counting and sampling. *Computational Complexity*, 15(4):298–341, 2006.

37 Ronen Shaltiel and Christopher Umans. Low-end uniform hardness versus randomness tradeoffs for AM. *SIAM J. Comput.*, 39(3):1006–1037, 2009.

38 Ronen Shaltiel and Emanuele Viola. Hardness amplification proofs require majority. *SIAM J. Comput.*, 39(7):3122–3154, 2010.

39 Michael Sipser. A complexity theoretic approach to randomness. In *STOC*, pages 330–335. ACM, 1983.

40 Larry J. Stockmeyer. The complexity of approximate counting (preliminary version). In *STOC*, pages 118–126. ACM, 1983.

41 Madhu Sudan, Luca Trevisan, and Salil P. Vadhan. Pseudorandom generators without the XOR lemma. *J. Comput. Syst. Sci.*, 62(2):236–266, 2001.

42 Luca Trevisan. Extractors and pseudorandom generators. *J. ACM*, 48(4):860–879, 2001.

43 Luca Trevisan and Salil P. Vadhan. Extracting randomness from samplable distributions. In *FOCS*, pages 32–42. IEEE Computer Society, 2000.

44 Christopher Umans. Pseudo-random generators for all hardnesses. *J. Comput. Syst. Sci.*, 67(2):419–440, 2003.

45 Andrew Chi-Chih Yao. Theory and applications of trapdoor functions (extended abstract). In *FOCS*, pages 80–91. IEEE Computer Society, 1982.

Learning Algorithms from Natural Proofs

Marco L. Carmosino[1], Russell Impagliazzo[2], Valentine Kabanets[3], and Antonina Kolokolova[4]

1 **Department of Computer Science, University of California, San Diego, USA**
mcarmosi@eng.ucsd.edu
2 **Department of Computer Science, University of California, San Diego, USA**
russell@eng.ucsd.edu
3 **School of Computing Science, Simon Fraser University, Burnaby, Canada**
kabanets@cs.sfu.ca
4 **Department of Computer Science, Memorial University of Newfoundland, St. John's, Canada**
kol@mun.ca

Abstract

Based on Håstad's (1986) circuit lower bounds, Linial, Mansour, and Nisan (1993) gave a quasipolytime learning algorithm for AC^0 (constant-depth circuits with AND, OR, and NOT gates), in the PAC model over the uniform distribution. It was an open question to get a learning algorithm (of any kind) for the class of $\mathsf{AC}^0[p]$ circuits (constant-depth, with AND, OR, NOT, and MOD_p gates for a prime p). Our main result is a quasipolytime learning algorithm for $\mathsf{AC}^0[p]$ in the PAC model over the uniform distribution with membership queries. This algorithm is an application of a general connection we show to hold between natural proofs (in the sense of Razborov and Rudich (1997)) and learning algorithms. We argue that a natural proof of a circuit lower bound against any (sufficiently powerful) circuit class yields a learning algorithm for the same circuit class. As the lower bounds against $\mathsf{AC}^0[p]$ by Razborov (1987) and Smolensky (1987) are natural, we obtain our learning algorithm for $\mathsf{AC}^0[p]$.

1998 ACM Subject Classification F.1.3 Complexity Measures and Classes

Keywords and phrases natural proofs, circuit complexity, lower bounds, learning, compression

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.10

1 Introduction

Circuit analysis problems, problems whose input or output is a Boolean circuit, are a crucial link between designing algorithms and proving lower bounds. For example, Williams [41, 43, 42] shows how to convert non-trivial Circuit-SAT algorithms into circuit lower bounds. In the other direction, there have been many circuit analysis algorithms inspired by circuit lower bound techniques [25, 4, 32, 34, 19, 20, 3, 8, 7, 33, 6, 9, 37], but outside the setting of derandomization [28, 2, 21, 18, 39, 23], few formal implications giving generic improvements.

Here we make a step towards such generic connections. While we are not able to show that an *arbitrary* way to prove circuit lower bounds yields circuit analysis algorithms, we show that any circuit lower bound proved through the general *natural proofs paradigm* of Razborov and Rudich [31] does yield such algorithms. Our main general result is the following.

▶ **Theorem 1.1** (Learning Algorithms from Natural Lower Bounds: Informal version)**.** *Natural proofs of circuit lower bounds imply learning algorithms for the same circuit class.*

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 10; pp. 10:1–10:24

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

Using known natural lower bounds [30, 35, 31], we get quasipolynomial-time learning algorithms for the hypothesis class $\mathsf{AC}^0[p]$, for any prime p (polynomial-size constant-depth circuits with AND, OR, NOT, and MOD_p gates).

▶ **Theorem 1.2** (Learning for $\mathsf{AC}^0[p]$: Simplified version)**.** *For every prime $p \geq 2$, there is a randomized algorithm that, given membership queries to an arbitrary n-variate Boolean function $f \in \mathsf{AC}^0[p]$, runs in quasi-polynomial time $n^{\mathsf{poly}\log n}$ and finds a circuit that computes f on all but $1/\mathsf{poly}(n)$ fraction of inputs.*

No learning algorithms for $\mathsf{AC}^0[p]$ were previously known. For AC^0, a learning algorithm was given by Linial, Mansour, and Nisan [25][1], based on Håstad's proof of strong circuit lower bounds for AC^0 [15].

We also apply the general result to immediately obtain the following compression algorithm, first developed (with somewhat stronger parameters) by Srinivasan [36].

▶ **Theorem 1.3** (Compression for $\mathsf{AC}^0[p]$: Simplified version)**.** *For every prime $p \geq 2$, there is a randomized algorithm that, given the 2^n-bit truth table of an arbitrary n-variate Boolean function $f \in \mathsf{AC}^0[p]$, runs in time $\mathsf{poly}(2^n)$ (polynomial in the input size), and outputs a circuit computing f of the circuit size at most 2^{n-n^μ}, for some $0 < \mu < 1$.*

1.1 Compression and learning algorithms from natural lower bounds

Informally, a *natural* lower bound for a circuit class Λ contains an efficient algorithm that distinguishes between the truth tables of "easy" functions (of low Λ-circuit complexity) and those of random Boolean functions. This notion was introduced by Razborov and Rudich [31] to capture a common feature of most circuit lower bound proofs: such proofs usually come with efficient algorithms that say something nontrivial about the structure of easy functions in the corresponding circuit class. In [31], this observation was used to argue that any circuit class with a natural lower bound is too weak to support cryptography: no strong pseudorandom generator can be computed by a small circuit from the class.

We show that natural circuit lower bounds also imply algorithms for compression and learning of Boolean functions from the same circuit class (provided the circuit class is not too weak). More precisely, we show how to reduce the task of compressing (learning) Boolean functions in a circuit class Λ to the task of distinguishing between the truth tables of functions of low Λ-circuit complexity and those of random functions. The latter task is exactly what is solved by an efficient algorithm embedded in any natural proof of Λ-circuit lower bounds.

Compression. Recall the compression task for Boolean functions: given the truth table of a Boolean function f, print a circuit that computes f. If f is unrestricted, the best guarantee for the circuit size is $2^n/n$ [26, 27], and such a circuit can be found in time $\mathsf{poly}(2^n)$, polynomial in the truth table size. We might however be able to do much better for restricted classes of functions. Let Λ be the set of functions computed by some circuit class Λ. Recent work has shown that we can "mine" specific lower bounds against Λ to compress functions $g \in \Lambda$ better than the universal construction [7]. This work suggests that there should be some generic connection between circuit lower bounds and compression algorithms, but such a connection was not known.

[1] Their algorithm works in a more general learning model without membership queries, but with access to labeled examples $(x, f(x))$ for uniformly random x.

We show that any circuit lower bound that is natural in the sense of Razborov and Rudich [31] yields a generic compression algorithm for Boolean functions from the same circuit class, provided the circuit class is sufficiently powerful (e.g., containing $\mathsf{AC}^0[p]$ for some prime $p \geq 2$).

A compression algorithm may be viewed as a special case of a natural property: if the compression fails, the function must have high complexity, and compression must fail for most functions. Thus we get an equivalence between these two notions for the case of randomized compression algorithms and BPP-computable natural properties. That is, for an appropriate circuit complexity class $\mathcal{C}$, a BPP-computable natural property against $\mathcal{C}$ implies the existence of a related $\mathcal{C}$-compression algorithm in BPP, and a $\mathcal{C}$-compression algorithm in BPP implies a BPP-computable natural property against $\mathcal{C}$. As our compression algorithms are randomized, we don't get such an equivalence for the case of deterministic natural properties.

Learning. The first stage of our algorithm is a lossy compression of the function in the sense that we get a small circuit that computes the function on *most* inputs. Because this first stage only examines the truth table of the function in relatively few locations, we can view this stage as a *learning algorithm*. This algorithm produces a circuit that approximately computes the given function f with respect to the uniform distribution, and uses membership queries to f. So it fits the framework of PAC learning for the uniform distribution, with membership queries.

Minimum Circuit Size Problem: Search to decision reduction. Our main result also yields a certain "search-to-decision" reduction for the Minimum Circuit Size Problem (MCSP). Recall that in MCSP, one is given the truth table of a Boolean function f, and a parameter s, and needs to decide if the minimum circuit size of f is less than s. Since an efficient algorithm for MCSP would make it a natural property (with excellent parameters), our main result implies the following. If MCSP is in BPP, then, given oracle access to any n-variate Boolean function f of circuit complexity s, one can find (in randomized polynomial time) a circuit of size $\mathsf{poly}(s)$ that computes f on all but $1/\mathsf{poly}(n)$ fraction of inputs.

1.2 Our proof techniques

One of the main tools we use is the Nisan-Wigderson (NW) generator construction [28]. Informally, this construction takes as input the truth table of a Boolean function f, and outputs an algorithm for the new function G_f mapping "short" input strings to "long" output strings. The function G_f is intended to be a *pseudo-random generator (PRG)* in the sense that no "small" Boolean circuit can "distinguish" the uniform distribution from the distribution of G_f's outputs (on uniformly random inputs to G_f). A circuit that can distinguish these two distribution is said to break the generator, and is called a *distinguisher*. Nisan and Wigderson [28] prove that if the initial function f has "high" circuit complexity, then the function G_f is indeed a PRG. Moreover, their proof is constructive in the sense that there is an efficient *reconstruction* algorithm that, given a distinguisher for G_f and oracle access to f, outputs a "small" Boolean circuit that approximately computes f. (See Section 2 for the formal definitions and statements.)

Intuitively, we can use this reconstruction algorithm as a *learning algorithm* for a Boolean function f in some circuit class Λ, provided we manage to find an efficient distinguisher for the NW generator G_f. As we shall argue, such a distinguisher for G_f is supplied by any natural proof of Λ-circuit lower bounds (natural property for the circuit class Λ)!

Thus, the main idea of our lossy-compression algorithm is, given the truth table of a Boolean function f from a circuit class Λ,

- imagine using f as the basis for the NW generator G_f,
- argue that the natural property R for the class Λ is a distinguisher for G_f,
- apply the reconstruction algorithm to R to produce a small circuit that approximates f.

For the described approach to work, we need to ensure that (1) there is an efficient reconstruction algorithm that takes a distinguisher for G_f and constructs a small circuit for (approximately computing) f, and (2) the natural property for Λ is a distinguisher for G_f.

For (1), we use the known efficient randomized algorithm that takes a distinguisher for G_f and constructs a small circuit approximately computing f, provided the algorithm is given oracle access to f. The existence of such a uniform algorithm was first observed by Impagliazzo and Wigderson [22] (based on [28, 2]) in the context of derandomizing BPP under uniform complexity assumptions. Simulating oracle access to f in the framework of [22] was quite nontrivial (and required the downward self-reducibility of f). In contrast, we are explicitly given the truth table of f (or allowed membership queries to f), and so oracle access to f is not an issue!

For (2), we must show that each output of the NW generator, when viewed as the truth table of a Boolean function, is computable by a small circuit from the circuit class for which we have a natural lower bound (and so the natural property algorithm can be used as a distinguisher to break the generator). Looking inside the construction of the NW generator, we note that, for a fixed seed (input) of G_f, each bit of the output of G_f is the value of f on some substring of the seed (chosen via a certain combinatorial structure, the NW design). We argue that the circuit complexity of the truth table output by the NW generator G_f is closely related to the circuit complexity of the original function f.

In particular, we show that if f is in $\mathsf{AC}^0[p]$, and the NW generator has exponential stretch (from $\mathsf{poly}(n)$ bits to 2^{n^γ} bits, for some $\gamma > 0$), then each string output by the NW generator is also a function in $\mathsf{AC}^0[p]$. If, on the other hand, we take the NW generator with certain polynomial stretch, we get that its output strings will be Boolean functions computable by $\mathsf{AC}^0[p]$ circuits of subexponential size. The trade-off between the chosen stretch of the NW generator and the circuit complexity of the string it outputs will be very important for the efficiency of our learning algorithms: the runtime of the learning algorithm will depend polynomially on the stretch of the NW generator. This makes our setting somewhat different than most applications of the NW generator. We will want to make the stretch as small as possible, but must set it above a threshold determined by the quantitative strength of the circuit lower bound that we start from. Thus, the *larger* the circuit size for which we have lower bounds, the *faster* the learning algorithms we get.

Note that if we break the NW generator based on a function f, we only get a circuit that agrees with f on slightly more than half of all inputs. To get a better approximation of f, we employ a standard "hardness amplification" encoding of f, getting a new, amplified function h, and then use h as the basis for the NW generator. The analysis of such hardness amplification is also constructive: it yields an efficient *reconstruction* algorithm that takes a circuit C_0 computing h on more than $1/2$ of the inputs, and constructs a new circuit C that computes the original f on most inputs.

For this amplification to work in our context, we need to ensure that the amplified function h is in the same circuit class as f, and is of related circuit complexity. We show that standard tools such as the Direct Product and XOR constructions have the required properties for $\mathsf{AC}^0[2]$. For $\mathsf{AC}^0[p]$ where p is prime other than 2, we can't use the XOR construction (as

PARITY cannot be computed in $\mathsf{AC}^0[p]$ for any prime $p > 2$ by Smolensky's lower bound [35]). We argue that the MOD_p function can be used for the required amplification within $\mathsf{AC}^0[p]$[2].

Thus, our actual lossy-compression algorithm for a circuit class Λ is as follows:

Given the truth table of a function $f \in \Lambda$,

1. Run the reconstruction algorithm for the NW generator G_h with the natural property for Λ as a distinguisher, where h is the amplified version of f. This produces a circuit C_0 computing h on more than $1/2$ of inputs.
2. Run the reconstruction algorithm for hardness amplification to get from C_0 a new circuit C that computes f on most inputs.

To turn this algorithm into an exact compression algorithm, we just patch up the errors by table lookup. Since there are relatively few errors, the size of the patched-up circuit will still be less that the trivial size $2^n/n$.

More interestingly, our lossy compression algorithm described above also yields a *learning* algorithm! The idea is that the reconstruction algorithm for the NW generator G_f runs in time polynomial in the size of the output of the generator, and so only needs at most that many oracle queries to the function f. Rather than being given the full truth table of f, such an algorithm can be simulated with just membership queries to f. Thus we get a learning algorithm with membership queries in the PAC model over the uniform distribution.

Since the runtime of this learning algorithm (and hence also the size of the circuit for f it produces) will be polynomial in the output length of the NW generator that we use to learn f, we would like to minimize the stretch of the NW generator[3]. However, as noted above, *shorter stretch* of the generator means *higher circuit complexity* of the truth table it outputs. This in turn means that we need a natural property that works for Boolean functions of higher circuit complexity (i.e., natural properties useful against large circuits). In the extreme case, to learn a polysize Boolean function f in polynomial time, we need to use the NW generator with polynomial stretch, and hence need a natural property useful against circuits of exponential size. In general, there will be a trade-off between the efficiency of our learning algorithm for the circuit class Λ and the usefulness of a natural circuit lower bound for Λ: the larger the size s such that a natural property is useful against Λ-circuits of size s, the more efficient the learning algorithm for Λ.

Razborov and Rudich [31] showed the $\mathsf{AC}^0[p]$ circuit lower bounds due to Razborov [30] and Smolensky [35] can be made into natural properties that are useful against circuits of weakly exponential size 2^{n^γ}, for some $\gamma > 0$ (dependent on the depth of the circuit). Plugging this natural property into our framework, we get our quasi-polynomial-time learning algorithm for $\mathsf{AC}^0[p]$, for any prime p.

We remark that our approach is quite similar to the way Razborov and Rudich [31] used natural properties to get new algorithms. They used natural properties to break the cryptographic pseudorandom function generator of [11], which by definition outputs functions of low circuit complexity. Breaking such a generator based on an assumed one-way function

[2] We stress that for our purposes it is important that the *forward direction* of the conditional PRG construction, from a given function f to a generator based on that f, be computable in some low nonuniform circuit class (such as $\mathsf{AC}^0[p]$). In contrast, in the setting of conditional derandomization, it is usually important that the *reverse direction*, from a distinguisher to a small circuit (approximately) computing the original function f, be computable in some low (nonuniform) circuit class (thereby contradicting the assumed hardness of f for that circuit class). One notable exception is hardness amplification within NP [29, 16, 38].

[3] This is in sharp contrast to the setting of derandomization where one wants to *maximize* the stretch of the generator, as it leads to a more efficient derandomization algorithm.

F leads to an efficient algorithm for inverting this function F well on average (contradicting the one-wayness of F). We, on the other hand, use the NW generator based on a given function f. The properties of the NW generator construction can be used to show that it outputs (the truth tables of) functions of low circuit complexity, relative to the circuit complexity of f. Thus a natural property for the appropriate circuit complexity class (with an appropriate size parameter) can be used to break this NW generator, yielding an efficient algorithm for producing a small circuit approximating f.

Discussion. One counter-intuitive development in the theory of pseudorandomness has been the prevalence of "win-win" arguments. Typically, in a win-win argument in pseudorandomness, one takes a construction of pseudorandom generator from a hardness assumption (such as the NW generator mentioned above) and applies it to a function that is *not known* to actually be hard. If the construction is still a PRG, that is a win; if it is not, one learns that the function in question is not hard, and perhaps finds a circuit computing it. Here, we take this paradigm one step further; ours is a "play-to-lose" argument. We apply the pseudorandom generator construction to a function f we *know* not to be hard, in such a way as to guarantee that the resulting generator G_f is *not* pseudorandom. The win in this argument is that the proof of the hardness to pseudorandomness connection gives a way of converting the non-randomness of the generator G_f into a way of computing f, thus translating the knowledge that f is easy to compute into an actual circuit computing f.

1.3 Related work

This work was prompted by results that circuit analysis algorithms imply circuit lower bounds. A natural question is: given that these algorithms are *sufficient* for circuit lower bounds, to what degree are they *necessary*? Apart from derandomization, no other equivalences between circuit analysis algorithms and circuit lower bounds are known. Some of the known circuit-analytic algorithmic tasks that would imply circuit lower bounds include: derandomization [18, 23, 1, 5], deterministic (lossy) compression or MCSP [7, 18], deterministic learning [10, 24], and deterministic (QBF) SAT algorithms [41, 33].

Bracketing the hardness vs. randomness setting, special cases of using circuit lower bounds to construct circuit analysis algorithms abound. Often, lower bounds are the *only* way that we know to construct these algorithms. Each of the following results uses the proof of a lower bound to construct an algorithm. The character and number of these results gives empirical evidence that there should be generic algorithms for circuit analysis based on generic lower bounds.

- Parity $\notin \mathsf{AC}^0 \rightsquigarrow \mathsf{AC}^0$-Learning [25], AC^0-SAT [19], and AC^0-Compression [7]
- $\mathrm{MOD}_q \notin \mathsf{AC}^0[p]$, p, q distinct primes, $\rightsquigarrow \mathsf{AC}^0[p]$-Compression [36]
- Andreev's function $\notin \mathrm{deMorgan}[n^{3-\epsilon}] \rightsquigarrow$ subcubic formula Compression [7]

All the lower bounds listed above belong to the natural proofs framework. Given these results, the obvious conjecture was that natural proofs imply some kind of generic circuit analysis algorithm. For instance, [7] suggested that every natural circuit lower bound should imply a compression algorithm. We take a step towards proving such an implication by showing that any natural circuit lower bound for a sufficiently powerful circuit class ($\mathsf{AC}^0[p]$ or bigger) does indeed lead to a randomized compression algorithm for the same circuit class.

The remainder of the paper. e give the necessary background in Section 2. Sections 3 and 4 summarize the useful properties of past constructions of black-box generators and black-box amplifications, which we revisit and modify to implement in $\mathsf{AC}^0[p]$. In Section 5, we use those tools to prove our main result that natural properties yield learning algorithms for circuit classes $\mathsf{AC}^0[p]$ and above, using a novel "play-to-lose" interpretation of pseudorandomness. On the other hand, in Section 6, we argue that our main result cannot be applied directly to AC^0 because the construction of Section 3 is impossible in AC^0. Section 7 contains concluding remarks and open questions.

2 Definitions and tools

2.1 Circuits and circuit construction tasks

For a circuit class Λ and a set of size functions $\mathcal{S}$, we denote by $\Lambda[\mathcal{S}]$ the set of $\mathcal{S}$-size n-input circuits of type Λ. When no $\mathcal{S}$ is explicitly given, it is assumed to be $\mathsf{poly}(n)$.

▶ **Definition 2.1** (Circuits (Approximately) Computing f)**.** Let $f : \{0,1\}^n \to \{0,1\}$ be some Boolean function, and let $\epsilon : \mathbb{N} \to [0,1]$ be an approximation bound. Then $\mathsf{CKT}_n(f)$ denotes the set of circuits that compute the function f on all n-bit inputs, and $\widetilde{\mathsf{CKT}}_n(f,\epsilon)$ the set of all circuits that compute f on all but an ϵ fraction of inputs.

▶ **Definition 2.2** (Circuit Builder Declarations (adapted from [22]))**.** Let A and B be indexed sets of circuits. A $T(n)$-*construction of* B *from* A is a probabilistic machine $\mathcal{M}(n,\alpha,A_n)$ which outputs a member of B_n with probability at least $1-\alpha$ in time $T(n)$, where the size of B_n is $\mathsf{poly}(|A_n|)$. We declare that such a machine exists by writing: $\textsc{Cons}(A \to B;\ T(n))$. Read this notation as "from A we can construct B in time $T(n)$." To assert the existence of a $T(n)$-*construction of* B *from* A*, with oracle* $\mathcal{O}$, where the machine $\mathcal{M}$ is equipped with an oracle for the language $\mathcal{O}$ but otherwise is as above, write: $\textsc{Cons}^{\mathcal{O}}(A \to B;\ T(n))$.

2.2 Learning and compression tasks

Let $f \in \Lambda$ be some Boolean function. The learner is allowed membership queries to f. That is, the learner may query an input $x \in \{0,1\}^n$ to the oracle, getting back the value $f(x)$.

▶ **Definition 2.3** (PAC learning over the uniform distribution with membership queries)**.** Let Λ be any class of Boolean functions. An algorithm A PAC-learns Λ if for any n-variate $f \in \Lambda$ and for any $\epsilon, \delta > 0$, given membership query access to f algorithm A prints with probability at least $1-\delta$ over its internal randomness a circuit $C \in \widetilde{\mathsf{CKT}}_n(f,\epsilon)$. The runtime of A is measured as a function $T = T(n, 1/\epsilon, 1/\delta, \mathrm{size}(f))$.

▶ **Definition 2.4** (Λ-Compression)**.** Given the truth table of n-variate Boolean function $f \in \Lambda$, print some Boolean circuit $C \in \mathsf{CKT}_n(f)$ computing f such that $|C| < 2^n/n$, the trivial bound.

▶ **Definition 2.5** (ϵ-Lossy Λ-Compression)**.** Given the truth table of n-variate Boolean function $f \in \Lambda$, print some Boolean circuit $C \in \widetilde{\mathsf{CKT}}_n(f,\epsilon)$ such that $|C| < 2^n/n$, the trivial bound.

The relevant parameters for compression are runtime and printed circuit size. We say that a compression algorithm is efficient if it runs in time $\mathsf{poly}(2^n)$, which is polynomial in the size of the truth-table supplied to the algorithm. Though we count any output circuit of size less than $2^n/n$ as a successful compression, we will of course want to optimize this. In previous

work, the size of the resulting circuits approximately matches the size of circuits for which we have lower bounds.

We remark that we do not obtain "proper" learning or compression: the output of the learning (compression) algorithm is an unrestricted circuit, not necessarily from the class to be learned (compressed).

2.3 Natural properties

Let F_n be the collection of all Boolean functions on n variables. Λ and Γ denote complexity classes. A combinatorial property is a sequence of subsets of F_n for each n.

▶ **Definition 2.6** (Natural Property [31]). A combinatorial property R_n is Γ-natural against Λ with density δ_n if it satisfies the following three conditions:

Constructivity: The predicate $f_n \stackrel{?}{\in} R_n$ is computable in Γ

Largeness: $|R_n| \geq \delta_n \cdot |F_n|$

Usefulness: For any sequence of functions f_n, if $f_n \in \Lambda$ then $f_n \notin R_n$, almost everywhere.

For each n, δ_n is a lower bound on the probability that $g \in F_n$ has R_n. The original definition in [31] sets $\delta_n \geq 2^{-O(n)}$. However, we show (see Lemma 2.7 below) that one may usually assume that $\delta_n \geq 1/2$. Note that in the wild, nearly all natural properties have δ_n close to one and $\Gamma \subseteq \mathsf{NC}^2$.

▶ **Lemma 2.7** (Largeness for natural properties). *Suppose P is a P-natural property of n-variate Boolean functions that is useful against class Λ of size $s(n)$, and has largeness $\delta_n \geq 2^{-cn}$, for some constant $c \geq 0$. Then there is another P-natural property P' that is useful against the class Λ of size $s'(n) := s(n/(c+1))$, and has largeness $\delta'_n \geq 1/2$.*

Proof. Define P' as follows:

> The truth table of a given $f : \{0,1\}^n \to \{0,1\}$ is in P' iff for at least one string $a \in \{0,1\}^k$, for $k = cn/(c+1)$, the restriction
>
> $$f_a(y_1, \ldots, y_{n-k}) := f(a_1, \ldots, a_k, y_1, \ldots, y_{n-k})$$
>
> is in P (as a function on $n-k = n/(c+1)$ variables).

Observe that testing P' on a given n-variate Boolean function f can be done in time $O(2^k) \cdot \mathsf{poly}(2^{n-k}) \leq \mathsf{poly}(2^n)$; so we have constructivity for P'. Next, if $f : \{0,1\}^n \to \{0,1\}$ has a Λ circuit of size less $s'(n)$, then each restricted subfunction $f_a : \{0,1\}^{n-k} \to \{0,1\}$ has a Λ circuit of size less than $s(n-k) \leq s'(n)$. Finally, a random function $f : \{0,1\}^n \to \{0,1\}$ yields 2^k independent random subfunctions, on $n-k$ variables each, and the probability that at least one of these $(n-k)$-variate functions satisfies P is at least $1 - (1 - 2^{-c(n-k)})^{2^k} = 1 - (1 - 2^{-k})^{2^k}$, which is at least $1/2$, as required. ◀

2.4 NW generator

▶ **Definition 2.8** (NW Design). For parameters $n, m, L \in \mathbb{N}$, a sequence of sets $S_1, \ldots, S_L \subseteq [m]$ is called an *NW design* if

- $|S_i| = n$, for all $1 \leq i \leq L$, and
- $|S_i \cap S_j| \leq \log L$, for all $1 \leq i \neq j \leq L$.

It is well-known that NW designs exist and can be efficiently constructed for any n, $m = O(n^2)$, and $L < 2^n$ [28]. In Section 3.1 below, we review the construction of NW designs from [28], and show that it can be implemented in $\mathsf{AC}^0[p]$ (Theorem 3.3). The efficiency of this construction of designs is necessary for our transfer theorem.

▶ **Definition 2.9** (NW Generator)**.** Let $f\colon \{0,1\}^n \to \{0,1\}$. For $m = n^2$ and a stretch function $L(m)\colon \mathbb{N} \to \mathbb{N}$, where $L(m) < 2^n$, let $S_1, \ldots, S_L \subseteq [m]$ be an NW design. Define the NW generator $G_f\colon \{0,1\}^m \to \{0,1\}^{L(m)}$ as:

$$G_f(z) = f(z|_{S_1})f(z|_{S_2})\ldots f(z|_{S_{L(m)}}), \tag{1}$$

where $z|_S$ denotes the $|S|$-length bit-string obtained by restricting z to the bit positions indexed by the set S.

Recall the notion of a distinguisher, a circuit that breaks a given generator.

▶ **Definition 2.10** (Distinguishers)**.** Let $L\colon \mathbb{N} \to \mathbb{N}$ be a stretch function, let $0 < \epsilon < 1$ be an error bound, and let $\mathcal{G} = \{g_m\colon \{0,1\}^m \to \{0,1\}^{L(m)}\}$ be a sequence of functions. Define $\mathsf{DIS}(\mathcal{G}, \epsilon)$ to be the set of all Boolean circuits D on $L(m)$-bit inputs satisfying:

$$\Pr_{z\in\{0,1\}^m}[D(g_m(z))] - \Pr_{y\in\{0,1\}^{L(m)}}[D(y)] > \epsilon.$$

▶ **Theorem 2.11** (NW Reconstruction [28, 22])**.** *We have*

$$\textsc{Cons}^f(\mathsf{DIS}(G_f, 1/5) \to \widetilde{\mathsf{CKT}}(f, 1/2 - 1/L(m));\ \mathsf{poly}(L(m))).$$

NW Reconstruction Algorithm. Since the reconstruction algorithm from the proof of Theorem 2.11 above is an essential ingredient in our learning algorithms, we sketch this algorithm below (omitting the correctness proof, which can be found in [28, 22]).

Let $G_f\colon \{0,1\}^m \to \{0,1\}^L$ be the NW generator based on a Boolean function $f\colon \{0,1\}^n \to \{0,1\}$, using the NW design $S_1, \ldots, S_L \subseteq [m]$. Suppose D is a distinguisher for G_f such that $D \in \mathsf{DIS}(G_f, 1/5)$. The following randomized algorithm will produce, with probability at least $1/\mathsf{poly}(L)$, a circuit C computing f on at least $1/2 + \Omega(1/L)$ fraction of inputs. It consists of a preprocessing stage, and a circuit construction stage.

Preprocessing

1. Pick a random $i \in [L]$.
2. For each $i \le j \le L$, fix the jth input of the distinguisher D to a random bit w_j.
3. For each $j \in [m] \setminus S_i$, fix the jth input of the generator G_f to a random bit z_j.
4. For each $1 \le j < i$, enumerate all $x \in \{0,1\}^n$ consistent with the partial assignment $z|_{S_j}$ from the previous step, query $f(x)$, and build the table T of pairs $(x, f(x))$.

Circuit construction

Using T, w_j's, and z_j's from preprocessing, build a circuit C following the template:
"On input $x \in \{0,1\}^n$,

1. Assign the inputs $z|_{S_i}$ of G_f to x, getting a fully specified input $z \in \{0,1\}^m$.
2. For each $1 \le j < i$, fix the jth input of D to $w_j = f(z|_{S_j})$, via table lookup in T.
3. If $D(w_1, \ldots, w_L) = 1$, then output w_i; otherwise, output $1 - w_i$."

To boost the probability of producing a good circuit C, we repeat the algorithm above $\mathsf{poly}(L)$ times, and estimate, using random sampling and membership queries to f, the agreement between f and each produced circuit C. We output the best circuit on our list.

3 Black-box generators

The main tool we need for our learning algorithms is a transformation, which we call *black-box generator*, taking a given function $f\colon \{0,1\}^n \to \{0,1\}$ to a family $G = \{g_z\}_{z\in I}$ of new Boolean functions $g_z\colon \{0,1\}^{n'} \to \{0,1\}$ satisfying the following properties:

Nonuniform Efficiency: Each function g_z has "small" circuit complexity relative to the circuit complexity of f.

Reconstruction: Any circuit distinguishing a random function g_z (for a uniformly random $z \in I$) from a random n'-variate Boolean function can be used (by an efficient randomized algorithm with oracle access to f) to construct a good approximating circuit for f.

Once we have such a black-box generator, we get our learning algorithm as follows. To learn a function $f\colon \{0,1\}^n \to \{0,1\}$, use the natural property as a distinguisher that rejects (the truth tables of) all functions g_z, $z \in I$, but accepts a constant fraction of truly random functions; apply the efficient reconstruction procedure to learn a circuit approximating f. Intuitively, we use the nonuniform efficiency property to argue that if f is an easy function in some circuit class Λ, then so is each function g_z, $z \in I$.

Next we give a more formal definition of a black-box generator. For a function f, we denote by Λ^f the class of oracle circuits in Λ that have f-oracle gates. Also recall that $\Lambda[s]$ denotes the class of Λ-circuits of size at most s.

▶ **Definition 3.1** (Black-Box (ϵ, L)-Generator Within Λ)**.** For a given error parameter $\epsilon\colon \mathbb{N} \to [0,1]$ and a stretch function $L\colon \mathbb{N} \to \mathbb{N}$, a *black-box* (ϵ, L)*-generator within* Λ is a mapping $\textsc{Gen}$ that associates with a given function $f\colon \{0,1\}^n \to \{0,1\}$ a family $\textsc{Gen}(f) = \{g_z\}_{z\in\{0,1\}^m}$ of Boolean functions $g_z\colon \{0,1\}^\ell \to \{0,1\}$, where $\ell = \log L(n)$, satisfying the following conditions for every $f : \{0,1\}^n \to \{0,1\}$:

Small Family Size: $m \le \mathsf{poly}(n, 1/\epsilon)$,

Nonuniform Λ-Efficiency: for all $z \in \{0,1\}^m$, $g_z \in \Lambda^f[\mathsf{poly}(m)]$, and

Reconstruction: $\textsc{Cons}^f(\mathsf{DIS}(\textsc{Gen}(f), 1/5) \to \widetilde{\mathsf{CKT}}(f,\epsilon); \mathsf{poly}(n, 1/\epsilon, L(n)))$, where we think of $\textsc{Gen}(f)$ as the distribution over the truth tables of functions $g_z \in \textsc{Gen}(f)$, for uniformly random $z \in \{0,1\}^m$.

We will prove the following.

▶ **Theorem 3.2.** *Let p be any prime. For every $\epsilon\colon \mathbb{N} \to [0,1]$ and $L\colon \mathbb{N} \to \mathbb{N}$ such that $L(n) \le 2^n$, there exists a black-box (ϵ, L)-generator within $\mathsf{AC}^0[p]$.*

We will use the NW generator as our black-box generator. For it to be within $\mathsf{AC}^0[p]$, we need NW designs to be computable within $\mathsf{AC}^0[p]$. We prove the following in the next subsection (see the proof of Theorem 3.7 in Section 3.1).

▶ **Theorem 3.3.** *Let p be any prime. There exists a constant $d_{MX} \ge 1$ such that, for any n and $L < 2^n$, there exists an NW design $S_1, \ldots, S_L \subseteq [m]$ with $m = O(n^2)$, each $|S_i| = n$, and $|S_i \cap S_j| \le \ell = \log L$ for all $1 \le i \ne j \le L$, such that the function $MX_{NW} : \{0,1\}^\ell \times \{0,1\}^m \to \{0,1\}^n$, defined by $MX_{NW}(i,z) = z|_{S_i}$, is computable by an $\mathsf{AC}^0[p]$ circuit of size $O(\ell \cdot n^3 \log n)$ and depth d_{MX}.*

Another ingredient we need for the proof of Theorem 3.2 is the following notion of black-box amplification. Let Λ be any circuit class.

▶ **Definition 3.4** (Black-Box (ϵ,δ)-Amplification within Λ)**.** For given $\epsilon, \delta > 0$, (ϵ,δ)*-amplification within* Λ is a mapping that associates with a given function $f\colon \{0,1\}^n \to \{0,1\}$ its *amplified* version, $\textsc{Amp}(f)\colon \{0,1\}^{n'} \to \{0,1\}$, satisfying the following conditions for every $f\colon \{0,1\}^n \to \{0,1\}$:

Short Input: $n' \leq \mathsf{poly}(n, 1/\epsilon, \log 1/\delta)$,
Nonuniform Λ-Efficiency: $\textsc{Amp}(f) \in \Lambda^f[\mathsf{poly}(n')]$,
Uniform P-Efficiency: $\textsc{Amp}(f) \in \mathsf{P}^f$, and
Reconstruction: $\textsc{Cons}^f(\widetilde{\mathsf{CKT}}(\textsc{Amp}(f), 1/2 - \delta) \to \widetilde{\mathsf{CKT}}(f, \epsilon);\ \mathsf{poly}(n, 1/\epsilon, 1/\delta))$.

We prove the following in the next section (see Theorems 4.3 and 4.8).

▶ **Lemma 3.5.** *Let p be any fixed prime. For all $0 < \epsilon, \delta < 1$, there is black-box (ϵ, δ)-amplification within $\mathsf{AC}^0[p]$.*

Now we are ready to prove Theorem 3.2.

Proof of Theorem 3.2. For a given n-variate Boolean function f, consider its amplified version $f^\star = (\epsilon(n), 1/L(n))\text{-}\textsc{Amp}(f)$, for the black-box amplification within $\mathsf{AC}^0[p]$ that exists by Lemma 3.5. We have that $f^\star$ is a function on $n' = \mathsf{poly}(n, 1/\epsilon, \log L) = \mathsf{poly}(n, 1/\epsilon)$ variables (using the assumption that $L(n) \leq 2^n$).

Let $G_{f^\star} \colon \{0,1\}^m \to \{0,1\}^{L(n)}$ be the NW generator based on the function $f^\star$, with the seed size $m = (n')^2$. Define $\textsc{Gen}(f) = \{g_z\}_{z \in \{0,1\}^m}$, where $g_z = G_{f^\star}(z)$. We claim that this $\textsc{Gen}(f)$ is an (ϵ, L)-black-box generator within $\mathsf{AC}^0[p]$. We verify each necessary property:

Small Family Size: $m = (n')^2 \leq \mathsf{poly}(n, 1/\epsilon)$.

Nonuniform $\mathsf{AC}^0[p]$-Efficiency: We know that $f^\star = \textsc{Amp}(f) \in (\mathsf{AC}^0[p])^f[\mathsf{poly}(m)]$. For each fixed $z \in \{0,1\}^m$, we have $g_z(i) = (G_{f^\star}(z))_i$, for $i \in \{0,1\}^\ell$, where $\ell = \log L(n)$. By the definition of the NW generator, $g_z(i) = f^\star(z|_{S_i})$. By Theorem 3.3, the restriction $z|_{S_i}$, as a function of z and i, is computable in $\mathsf{AC}^0[p]$ of size $\mathsf{poly}(n')$ and some fixed depth d_{MX}. It follows that each g_z is computable in $(\mathsf{AC}^0[p])^f[\mathsf{poly}(m)]$.

Reconstruction: The input to reconstruction is $D \in \mathsf{DIS}(G_{\textsc{Amp}(f)}, 1/5)$. Let $\mathcal{M}_{NW}$ be the reconstruction machine from the NW construction, and let $\mathcal{M}_{\textsc{Amp}}$ be the reconstruction machine from $(\epsilon, 1/L)$-amplification. We first run $\mathcal{M}_{NW}^{\textsc{Amp}(f)}(D)$ to get, in time $\mathsf{poly}(L)$, a circuit $C \in \widetilde{\mathsf{CKT}}(\textsc{Amp}(f), 1/2 - 1/L(n))$; note that we can provide this reconstruction algorithm oracle access to $\textsc{Amp}(f)$, since $\textsc{Amp}(f) \in \mathsf{P}^f$ by the uniform P-efficiency property of black-box amplification. Next we run $\mathcal{M}_{\textsc{Amp}}^f$ on C to get $C' \in \widetilde{\mathsf{CKT}}(f, \epsilon)$, in time $\mathsf{poly}(n, 1/\epsilon, L(n))$. ◀

3.1 NW designs in $\mathsf{AC}^0[p]$

Here we show that the particular NW designs we need in our compression and learning algorithms can be constructed by small $\mathsf{AC}^0[p]$ circuits, for any fixed prime p. Consider an NW design $S_1, \ldots, S_L \subseteq [m]$, for $m = O(n^2)$, where

- each set S_i is of size n,
- the number of sets is $L = 2^\ell$ for $\ell \leq n$, and
- for any two distinct sets S_i and S_j, $i \neq j$, we have $|S_i \cap S_j| \leq \ell$.

We show a particular construction of such a design that has the following property: the index set $[m]$ is partitioned into n disjoint subsets $U_1, \ldots, U_n$ of equal size $(m/n) \in O(n)$. For each $1 \leq i \leq L$, the set S_i contains exactly one element from each subset U_j, over all $1 \leq j \leq n$. For $1 \leq j \leq n$ and $1 \leq k \leq O(n)$, we denote by $(U_j)_k$ the kth element in the subset U_j.

To describe such a design, we use the following Boolean function g: for $1 \le i \le L$, $1 \le j \le n$, and $1 \le k \le O(n)$, we define $g(i,j,k) = 1$ iff $(U_j)_k \in S_i$. We will prove the following.

▶ **Theorem 3.6.** *There exists a constant $d_{NW} \ge 1$ such that, for any prime p, there exists a family of functions $g : \{0,1\}^{\ell+2\log n} \to \{0,1\}$ that are the characteristic functions for some NW design with the parameters as above, so that $g \in \mathsf{AC}^0[p]$ of size $O(n^2 \log n)$ and depth d_{NW}.*

Proof. Recall the standard construction of NW designs from [28]. Let F be a field of size $O(n)$. Consider an enumeration of L polynomials of degree at most ℓ over F, with all coefficients in $\{0,1\}$; there are at least $2^\ell = L$ such polynomials. We associate each such polynomial with a binary string $i = i_1 \dots i_\ell \in \{0,1\}^\ell$, so that i corresponds to the polynomial

$$A_i(x) = \sum_{j=1}^{\ell} i_j \cdot x^{j-1}$$

over the field F. Let $r_1, \dots, r_{|F|}$ be some canonical enumeration of the elements of F. For each binary string $i \in \{0,1\}^\ell$, we define a set $S_i = \{(r_j, A_i(r_j)) \mid 1 \le j \le n\}$. Note that $|S_i| = n$, and S_i defines a set of n pairs in the universe $F \times F$ of $O(n^2)$ pairs (hence the universe size for this construction is $O(n^2)$). Finally, any two distinct degree $(\ell-1)$ polynomials $A_i(x)$ and $A_j(x)$ may agree on at most ℓ points $r \in F$, and so we have $|S_i \cap S_j| \le \ell$ for the sets S_i and S_j, corresponding to the polynomials $A_i(x)$ and $A_j(x)$.

Arrange the elements of the universe $[m]$ on an $n \times (m/n)$ grid. The n rows of the grid are indexed by the first n field elements $r_1, \dots, r_n$, and the columns by all fields elements $r_1, \dots, r_|F|$. For each j, $1 \le j \le n$, define U_j to be the elements of $[m]$ that belong to the row j of the grid. We get that every set $S_i = \{(r_j, A_i(r_j)) \mid 1 \le j \le n\}$ picks exactly one element from each of the n sets $U_1, \dots, U_n$.

We will argue that this particular design construction is computable in $\mathsf{AC}^0[p]$ of size polynomial in ℓ, for each prime p. Let p be any fixed prime (which we think of as a constant). Let F be an extension field over $\mathbf{GF}(p)$ of the least size so that $|F| \ge n$; such a field is described by some polynomial over $\mathbf{GF}(p)$ of degree $O(\log_p n)$, and is of size at most $pn = O(n)$. As before, let $r_1, \dots, r_{|F|}$ be a canonical enumeration of the field elements in F.

Define the following $n \times \ell$ matrix M: for $1 \le j \le n$ and $1 \le k \le \ell$, we have $M_{j,k} = (r_j)^k$, where the power $(r_j)^k$ is computed within the field F. Then the values $A_i(r_1), \dots, A_i(r_n)$ may be read off from the column vector obtained by multiplying the matrix M by the column vector $i \in \{0,1\}^\ell$, in the field F. For a particular $1 \le j \le n$, we have $A_i(r_j) = \sum_{k=1}^{\ell} M_{j,k} \cdot i_k$. Since each $i_k \in \{0,1\}$, the latter reduces to the task of adding a subset of ℓ field elements. Each field element of F is a polynomial over $\mathbf{GF}(p)$ of degree $k \le O(\log n)$, and so adding a collection of elements from F reduces to the coordinate-wise summation modulo p of k-element vectors in $(\mathbf{GF}(p))^k$. The latter task is easy to do in $\mathsf{AC}^0[p]$[4].

For any $1 \le i \le L$, $1 \le j \le n$, and $1 \le k \le |F|$, $g(i,j,k) = 1$ iff $A_i(r_j) = r_k$. To compute $g(i,j,k)$, we need to evaluate the polynomial $A_i(x)$ at r_j, and then check if the result is

[4] We code elements of $\mathbf{GF}(p)$ by p-wire bundles, where wire i is on iff the bundle codes the ith element of $\mathbf{GF}(p)$. An addition, multiplication, or inverse in the field $\mathbf{GF}(p)$ can be implemented in AC^0. To add up a tuple of field elements, we first convert each field element from the representation above to the unary representation (using constant-depth selection circuits). Then we lead these unary encodings into a layer of p gates, $\oplus_p^j$, for $0 \le j \le p-1$, where $\oplus_p^j$ is the gate $\oplus_p$ with $p-j$ extra inputs 1. Thus the gate $\oplus_p^j$ on inputs $x_1, \dots, x_n \in \mathbf{GF}(p)$ outputs 1 iff $x_1 + \dots + x_n = j \mod p$. Note that exactly one of the gates $\oplus_p^j$ will output 1, giving us the desired field element in our encoding.

equal to r_k. To this end, we "hard-code" the matrix M into the circuit (which incurs the cost at most $O(n\ell \log n)$ bits of advice). We compute $A_i(r_j)$ by computing the matrix-vector product $M \cdot i$, and restricting to the jth coordinate of the resulting column vector. This computation involves $O(\log n)$ summations of ℓ field elements of $\mathbf{GF}(p)$ modulo p, over n rows of the matrix M. The resulting field element is described an $O(\log n)$-element vector of elements from the underlying field $\mathbf{GF}(p)$. Using $O(\log n)$ operations over $\mathbf{GF}(p)$, we can check if this vector equals the vector corresponding to r_k.

It is easy to see that this computation can be done in some fixed constant depth d_{NW} by an $\mathsf{AC}^0[p]$ circuit of size $O(\ell \cdot n \log n)$, which can be bounded by $O(n^2 \log n)$, as required. ◀

As a corollary, we get Theorem 3.3, which we re-state below.

▶ **Theorem 3.7.** *Let p be any prime. There exists a constant $d_{MX} \geq 1$ such that, for any n and $L < 2^n$, there exists an NW design $S_1, \ldots, S_L \subseteq [m]$ with $m = O(n^2)$, each $|S_i| = n$, and $|S_i \cap S_j| \leq \ell = \log L$ for all $1 \leq i \neq j \leq L$, such that the function $MX_{NW} : \{0,1\}^\ell \times \{0,1\}^m \to \{0,1\}^n$, defined by $MX_{NW}(i,z) = z|_{S_i}$, is computable by an $\mathsf{AC}^0[p]$ circuit of size $O(\ell \cdot n^3 \log n)$ and depth d_{MX}.*

Proof. Let $g(i,j,k)$ be the characteristic function for the NW design from Theorem 3.6, where $|i| = \ell$, $|j| = \log n$, and $|k| = \log n + \log c$, for some constant $c \geq 1$. We have $g \in \mathsf{AC}^0[p]$ of size $O(\ell \cdot n \log n)$ and depth d_{NW}. Let $U_1, \ldots, U_n \subseteq [m]$ be the sets of size cn each that partition $[m]$ so that every S_i contains exactly one element from every U_j, $1 \leq j \leq n$.

Let $i_1, \ldots, i_\ell$ and $z_1, \ldots, z_m$ denote the input gates of MX_{NW}, and let $y_1, \ldots, y_n$ denote its output gates. Associate each gate y_j with the set U_j of indices in $[m]$, for $1 \leq j \leq n$. For each $1 \leq i \leq L$ and each $1 \leq j \leq n$, define

$$y_j = \vee_{k=1}^{cn} \; g(i,j,k) \wedge (z|_{U_j})_k.$$

Clearly, the defined circuit computes MX_{NW}. It has size $O(\ell \cdot n^3 \log n)$ and depth $d_{MX} \leq d_{NW} + 2$, as required. ◀

Let G_f be the NW generator based on a function f, using the NW design $S_1, \ldots, S_L$ from Theorem 3.3. For each fixed seed z, define the function $g_z : \{0,1\}^\ell \to \{0,1\}$, for $\ell = \log L$, as $g_z(i) = (G_f(z))_i = f(z|_{S_i})$, where $1 \leq i \leq L$. By Theorem 3.3, we get $g_z \in (\mathsf{AC}^0[p])^f$. See Figure 1 for the description of a small circuit for g_z that combines the $\mathsf{AC}^0[p]$ circuit for MX_{NW} with a circuit for f.

4 Black-box amplification

Here we show that black-box amplification (Definition 3.4) is possible within $\mathsf{AC}^0[p]$, for any prime $p \geq 2$, as required for the proof that black-box generators within $\mathsf{AC}^0[p]$ exist (Theorem 3.2). For $\mathsf{AC}^0[2]$, we shall use standard hardness amplification tools from pseudorandomness: Direct Product and XOR construction. For $\mathsf{AC}^0[p]$, $p \neq 2$, we will need to use something else in place of XOR, as small $\mathsf{AC}^0[p]$ circuits can't compute Parity [35]. We will replace XOR with a MOD_p function, also using an efficient conversion from $\{0,1,\ldots,p-1\}$-valued functions to Boolean functions, which preserves the required amplification parameters.

For a Boolean function $f : \{0,1\}^n \to \{0,1\}$ and a parameter $k \in \mathbb{N}$, the k-wise direct product of f is $f^k : \{0,1\}^{nk} \to \{0,1\}^k$, where $f^k(x_1, \ldots, x_k) = (f(x_1), \ldots, f(x_k))$ for $x_i \in \{0,1\}^n$, $1 \leq i \leq k$. It is well-known that the Direct Product (DP) construction amplifies hardness of a given function f in the sense that a circuit somewhat nontrivially

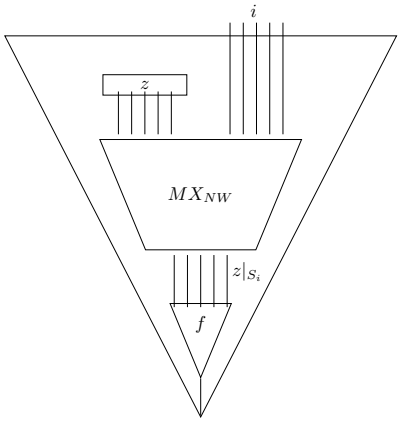


Figure 1 A circuit for $g_z(i) = f(z|_{S_i})$.

approximating the function f^k may be used to get a new circuit that approximates the original function f quite well [13], and, moreover, this new circuit for f can be constructed efficiently uniformly [22]. We shall use the following algorithm due to [17] that has optimal parameters (up to constant factors).

▶ **Theorem 4.1** (DP Reconstruction [17]). *There is a constant c and a probabilistic algorithm $\mathcal{A}$ with the following property. Let $k \in \mathbb{N}$, and let $0 < \epsilon, \delta < 1$ be such that $\delta > e^{-\epsilon k/c}$. For a Boolean function $f\colon \{0,1\}^n \to \{0,1\}$, let C' be any circuit in $\widetilde{\mathsf{CKT}}(f^k, 1-\delta)$. Given such a circuit C', algorithm $\mathcal{A}$ outputs with probability $\Omega(\delta)$ a circuit $C \in \widetilde{\mathsf{CKT}}(f, \epsilon)$.*

DP Reconstruction Algorithm. The algorithm $\mathcal{A}$ in Theorem 4.1 is a uniform randomized NC^0 algorithm (with one C'-oracle gate), and the produced circuit C is an AC^0 circuit of size $\mathsf{poly}(n, k, \log 1/\epsilon, 1/\delta)$ (with $O((\log 1/\epsilon)/\delta)$ of C'-oracle gates). We sketch this algorithm below. It consists of a preprocessing stage and a circuit construction stage. For simplicity, we allow the constructed circuit to be randomized; it can easily be made deterministic by choosing all required randomness in the preprocessing stage.

Preprocessing

Randomly pick a set B_0 of k strings in $\{0,1\}^n$. Pick a random subset $A \subset B_0$ of size $k/2$. Evaluate C' on a k-tuple $\vec{b_0}$ that is a random permutation of the strings in B_0, and note the answers $\vec{a}$ given by $C'(\vec{b_0})$ for the strings in A.

Circuit construction

Using A and $\vec{a}$ from preprocessing, build a randomized circuit C following the template:
"On input $x \in \{0,1\}^n$, if $x \in A$, then output the corresponding answer in $\vec{a}$. Otherwise, for $m = O((\log 1/\epsilon)/\delta)$ times,
1. sample a random k-set B such that $A \cup \{x\} \subset B$;
2. evaluate C' on a k-tuple $\vec{b}$ that is a random permutation of the strings in B;
3. if the answers of $C'(\vec{b})$ for A are consistent with $\vec{a}$, then output $C'(\vec{b})_x$ (the answer C' gave for x), and stop.
If no output is produced after m iterations, output a random bit."

Next, we need to convert a non-Boolean function $f^k\colon \{0,1\}^{kn} \to \{0,1\}^k$ into a Boolean function h such that a circuit approximately computing h would uniformly efficiently yield

a circuit approximately computing f^k, where the quality of approximation is essentially preserved. To this end, we "collapse" the k-bit output of f^k to a single number modulo a prime p, using the Goldreich-Levin construction [12] over $F = \mathbf{GF}(p)$: For $g\colon \{0,1\}^m \to \{0,1\}^k$, define $g^{GL}\colon \{0,1\}^m \times F^k \to F$ to be

$$g^{GL}(x_1,\ldots,x_m,r_1,\ldots,r_k) = \sum_{i=1}^{k} r_i \cdot g(x_1,\ldots,x_m)_i,$$

where all arithmetic is over the field F.

We will describe an efficient reconstruction algorithm that takes a circuit computing the function g^{GL} on more than $1/p+\gamma$ fraction of inputs, for some $\gamma > 0$, and produces a circuit that computes g on more than $\Omega(\gamma^3)$ fraction of inputs. The main ingredient of this algorithm is the following result first proved by Goldreich and Levin [12] for the case of $p = 2$, and later generalized by Goldreich, Rubinfeld, and Sudan [14] to all primes p.

▶ **Theorem 4.2** (GL Reconstruction [12, 14]). *There is a probabilistic algorithm $\mathcal{A}$ with the following property. Let $h \in F^k$ be arbitrary, and let $B\colon F^k \to F$ be such that $\mathbf{Pr}_{r\in F^k}[B(r) = \langle h, r\rangle] \geq 1/p+\gamma$ for some $\gamma > 0$, where $\langle x, y\rangle = \sum_{i=1}^{k} x_i \cdot y_i \mod p$. Then, given oracle access to B and the parameter γ, the algorithm $\mathcal{A}$ runs in time $\mathsf{poly}(k, 1/\gamma)$ and outputs a list of size $O(1/\gamma^2)$ such that, with probability at least $1/2$, the tuple h is on the list.*

GL Reconstruction Algorithm. We sketch below the algorithm $\mathcal{A}$ of Theorem 4.2.

Proceed in k rounds, maintaining after round i a list $\mathcal{H}_i$ of length-i tuples in F^i; the list after round k is the final output. In round i:

1. Extend each tuple in $\mathcal{H}_{i-1}$ by one element in all $|F|$ possible ways.
2. For each extended tuple $\vec{c} \in F^i$, include $\vec{c}$ in $\mathcal{H}_i$ iff it passes the following test:

   Randomly pick $m = \mathsf{poly}(k/\gamma)$ tuples $\vec{s}_1,\ldots,\vec{s}_m \in F^{k-i}$. For each $\vec{s}_i$ and each $\sigma \in F$, estimate $\mathbf{Pr}_{\vec{r}\in F^i}[B(\vec{r},\vec{s}) = \langle \vec{c},\vec{r}\rangle + \sigma]$. If at least one of these estimates is significantly larger than $1/p$, then accept; otherwise, reject.

4.1 Case of $\mathsf{AC}^0[2]$

Theorems 4.1 and 4.2 imply the following.

▶ **Theorem 4.3** (Black-Box Amplification within $\mathsf{AC}^0[2]$). *For any $0 < \epsilon, \gamma < 1$, there is black-box (ϵ,γ)-amplification within $\mathsf{AC}^0[2]$.*

Proof. Given $f\colon \{0,1\}^n \to \{0,1\}$ in $\mathsf{AC}^0[2]$ of size s, and given $0 < \epsilon, \delta < 1$, define $\textsc{Amp}(f)$ as follows:

1. Set $k = \lceil (3c)\cdot 1/\epsilon \cdot \ln 1/\gamma\rceil + 1$, where c is the constant in Theorem 4.1.
2. Define g to be the direct product $f^k\colon \{0,1\}^{nk} \to \{0,1\}^k$.
3. Define $\textsc{Amp}(f)$ to be $g^{GL}\colon \{0,1\}^{nk+k} \to \{0,1\}$ over $F = \mathbf{GF}(2)$.

▶ **Claim 4.4.** *For any $\gamma > 0$, we have*

$$\textsc{Cons}^f(\widetilde{\mathsf{CKT}}(g^{GL}, 1/2-\gamma) \to \widetilde{\mathsf{CKT}}(g, 1-\Omega(\gamma^3)); \mathsf{poly}(n,k,1/\gamma)).$$

Proof. Suppose we are given a circuit $C' \in \widetilde{\mathsf{CKT}}(g^{GL}, 1/2-\gamma)$. Let $\mathcal{A}_{GL}$ be the Goldreich-Levin algorithm of Theorem 4.2. Consider the following algorithm $\mathcal{A}_1$ that attempts to compute g:

For a given input $x \in \{0,1\}^{nk}$, define a circuit $B_x(r) := C'(x,r)$, for $r \in \{0,1\}^k$. Run $\mathcal{A}_{GL}$ on B_x, with parameter $\gamma/2$, getting a list L of k-bit strings. Output a uniformly random k-bit string from the list L.

Correctness Analysis of $\mathcal{A}_1$: By averaging, for each of at least $\gamma/2$ fraction of strings $x \in \{0,1\}^{nk}$, the circuit $B_x(r) := C'(x,r)$ agrees with $g^{GL}(x,r) = \langle g(x), r\rangle$ on at least $1/2 + \gamma/2$ fraction of strings $r \in \{0,1\}^k$. For each such x, the circuit B_x satisfies the condition of Theorem 4.2, and so the GL algorithm will find, with probability at least $1/2$, a list L of $O(1/\gamma^2)$ strings in $\{0,1\}^k$ that contains the string $g(x)$. Conditioned on the list containing the string $g(x)$, if we output a random string on that list, we get $g(x)$ with probability at least $1/|L| \geq \Omega(\gamma^2)$. Overall, the fraction of inputs x where $\mathcal{A}_1$ correctly computes $g(x)$ is at least $\frac{\gamma}{2} \cdot \frac{1}{2} \cdot \Omega(\gamma^2) \geq \Omega(\gamma^3)$. The runtime of $\mathcal{A}_1$ is $\mathsf{poly}(|C'|, k, n, 1/\gamma)$. ◀

By Theorem 4.1, we have

$$\textsc{Cons}^f(\widetilde{\mathsf{CKT}}(f^k, 1-\mu) \to \widetilde{\mathsf{CKT}}(f,\epsilon); \mathsf{poly}(n,k,\log 1/\epsilon, 1/\mu)),$$

as long as $\mu > e^{-\epsilon k/c}$, for some fixed constant $c > 0$. Combining this with Claim 4.4 yields

$$\textsc{Cons}^f(\widetilde{\mathsf{CKT}}(\textsc{Amp}(f), 1/2-\gamma) \to \widetilde{\mathsf{CKT}}(f,\epsilon); \mathsf{poly}(n,1/\epsilon,1/\gamma)),$$

as long as $\gamma^3 > e^{-\epsilon k/c}$, which is equivalent to $\gamma > e^{-\epsilon k/c'}$, for $c' = 3c$. Our choice of k satisfies this condition.

Finally, we verify that $\textsc{Amp}(f)$ also satisfies the other conditions of black-box amplification:

- $(f^k)^{GL}$ is defined on inputs of size $kn + k \leq O(n \cdot 1/\epsilon \cdot \log 1/\gamma)$.
- If $f \in \mathsf{AC}^0[2]$ of size s, then f^k is in $\mathsf{AC}^0[2]$ of size $O(s \cdot k) = O(s \cdot 1/\epsilon \cdot \log 1/\gamma)$, and $(f^k)^{GL}$ is of size at most the additive term $O(k)$ larger.
- $(f^k)^{GL}$ is in P^f.

Thus, $\textsc{Amp}(f)$ defined above is black-box (ϵ,γ)-amplification of f, as required. ◀

4.2 Case of $\mathsf{AC}^0[p]$ for primes $p > 2$

For $\mathsf{AC}^0[p]$ circuits, with $p > 2$, we can't use the XOR construction above, as Parity is not computable by small $\mathsf{AC}^0[p]$ circuits [35]. A natural idea to amplify a given function f is to apply the Goldreich-Levin construction g^{GL} over the field $F = \mathbf{GF}(p)$ to the direct-product function $g = f^k$, for an appropriate value of k. Theorem 4.2 guarantees that if we have a circuit that computes g^{GL} on more than $1/p + \gamma$ fraction of inputs, then we can efficiently construct a circuit that computes g on $\Omega(\gamma^3)$ fraction of inputs; the proof is identical to that of Claim 4.4 inside the proof of Theorem 4.3 for the case of $\mathsf{AC}^0[2]$ above.

The only problem is that the function g^{GL} defined here is *not* Boolean-valued, but we need a Boolean function to plug into the NW generator in order to complete our construction of a black-box generator within $\mathsf{AC}^0[p]$. We need to convert g^{GL} into a Boolean function h in such a way that if h can be computed by some circuit on at least $1/2 + \mu$ fraction of inputs, then g^{GL} can be computed by a related circuit on at least $1/p + \mu'$ fraction of inputs, where μ and μ' are close to each other.

We use von Neumann's idea for converting a coin of unknown bias into a perfectly unbiased coin [40]. Given a coin that is "heads" with some (unknown) probability $0 < p < 1$, flip the coin twice in a row, independently, and output 0 if the trials were ("heads", "tails"), or 1 if the trials were ("tails", "heads"). In case both trials came up the same (i.e., both "heads", or both "tails"), flip the coins again.

Observe that, conditioned on producing an answer $b \in \{0,1\}$, the value b is uniform over $\{0,1\}$ (as both conditional probabilities are equal to $p(1-p)/(1-p^2-(1-p)^2)$. The probability of not producing an answer in one attempt is $p^2+(1-p)^2$, the collision probability of the distribution $(p, 1-p)$. If p is far away from 0 and 1, the probability that we need to repeat the flipping for more than t trials diminishes exponentially fast in t.

In our case, we can think of the value of g^{GL} on a uniformly random input as a distribution over F. Assuming that this distribution is close to uniform over F, we will define a new Boolean function h based on g^{GL} so that the output of h on a uniformly random input is close to uniform over $\{0,1\}$. Our analysis of h will be constructive in the following sense. If we are given a circuit that distinguishes the distribution of the outputs of h from uniform, then we can efficiently construct a circuit that distinguishes the distribution of the outputs of g^{GL} from uniform over F. Finally, using the standard tools from pseudorandomness (converting distinguishers into predictors), we will efficiently construct from this distinguisher circuit a new circuit that computes g^{GL} on noticeably more than $1/p$ fraction of inputs.

The construction of this function h follows the von Neumann trick above. Formally we have the following.

▶ **Definition 4.5** (von Neumann trick function)**.** For an integer parameter $t > 0$, define the function $E^{vN}\colon (F^2)^t \to \{0,1\}$ as follows: For pairs $(a_1,b_1),\ldots,(a_t,b_t) \in F \times F$, set

$$E^{vN}((a_1,b_1),\ldots,(a_t,b_t)) = \begin{cases} 1 & \text{if, for each } 1 \le i \le t,\ a_i = b_i \\ 1 & \text{if } (a_i,b_i) \text{ is the first unequal pair and } a_i > b_i \\ 0 & \text{if } (a_i,b_i) \text{ is the first unequal pair and } a_i < b_i \end{cases}$$

It is not hard to see that E^{vN} is computable in AC^0. Moreover, for independent uniformly distributed inputs, the output of E^{vN} is a random coin flip, with bias at most $(1/p)^t$.

▶ **Claim 4.6.** *Let $\mathcal{F}$ be the uniform distribution over the field $F = \mathbf{GF}(p)$, and let $\mathcal{G} = (\mathcal{F}^2)^t$ be the uniform distribution over sequences of t pairs of elements from F. Then $\left|\mathbf{Pr}_{r\in\mathcal{G}}[E^{vN}(r)=1] - \mathbf{Pr}_{r\in\mathcal{G}}[E^{vN}(r)=0]\right| \le p^{-t}$.*

Proof. Conditioned on having some unequal pair in the sample from $\mathcal{G}$, the bias of the random variable $E^{vN}(\mathcal{G})$ is 0. Conditioned on having no such unequal pair, the bias is at most 1. Note that the collision probability of the uniform distribution over $\mathbf{GF}(p)$ is $\sum_{i=1}^{p} p^{-2} = p^{-1}$. So the probability of having collisions in all t independent samples from $\mathcal{F}^2$ is p^{-t}. Thus, the overall bias is at most p^{-t}. ◀

Next, given $g^{GL}\colon D \to F$, for the domain $D = \{0,1\}^m \times F^k$, define $h^{vN}\colon (D^2)^t \to \{0,1\}$ as follows:

$$h^{vN}((a_1,b_1),\ldots,(a_t,b_t)) = E^{vN}((g^{GL}(a_1),g^{GL}(b_1)),\ldots,(g^{GL}(a_t),g^{GL}(b_t))).$$

▶ **Theorem 4.7.** *For any $0 < \mu < 1$ and $1 > \gamma > \Omega(\mu/(\log 1/\mu))$, we have*

$$\textsc{Cons}^f(\widetilde{\mathsf{CKT}}(h^{vN}, 1/2-\mu) \to \widetilde{\mathsf{CKT}}(g^{GL}, 1-1/p-\gamma);\ \mathsf{poly}(k,m,\mathsf{poly}(1/\mu))).$$

Proof. Recall some basic definition from pseudorandomness theory. We say that distributions X and Y are computationally (η,s)-indistinguishable, denoted by $X \overset{\eta,s}{\approx} Y$ if, for any circuit T of size s, the probability that T accepts a sample from X is the same as the probability T accepts a sample from Y, to within $\pm\eta$.

We want to show that if h^{vN} is predictable with probability better than $1/2$, then g^{GL} is predictable with probability better than $1/p$. We will argue the contrapositive: suppose g^{GL} is unpredictable, and show that h^{vN} is unpredictable. This will take a sequence of steps.

Let $\mathcal{D}$ denote the uniform distribution over D, $\mathcal{F}$ the uniform distribution over F, and $\mathcal{U}$ the uniform distribution over $\{0,1\}$. Assume g^{GL} is unpredictable by circuits of size s with probability better than $1/p+\gamma$, for some $\gamma > 0$. This implies the following sequence of statements:

1. $(\mathcal{D}, g^{GL}(\mathcal{D})) \overset{2\gamma,\Omega(s)}{\approx} (\mathcal{D},\mathcal{F})$ (unpredictable $\Rightarrow$ indistinguishable)
2. $(\mathcal{D}^{2t}, g^{GL}(\mathcal{D})^{2t}) \overset{4t\gamma,\Omega(s/t)}{\approx} (\mathcal{D}^{2t}, F^{2t})$ (hybrid argument)
3. $(\mathcal{D}^{2t}, E^{vN}(g^{GL}(\mathcal{D})^{2t})) \overset{4t\gamma,\Omega((s/t)-\mathsf{poly}(t))}{\approx} (\mathcal{D}^{2t}, E^{vN}(F^{2t}))$ (applying h^{vN})
4. $(\mathcal{D}^{2t}, h^{vN}(\mathcal{D}^{2t})) \overset{4t\gamma+p^{-t},\Omega((s/t)-\mathsf{poly}(t))}{\approx} (\mathcal{D}^{2t},\mathcal{U})$ (by Claim 4.6)

Finally, the last statement implies (via the "indistinguishable to unpredictable" direction) that h^{vN} cannot be computed on more than $1/2+\mu$ fraction of inputs by any circuit of size $\Omega((s/t)-\mathsf{poly}(t))$, where $\mu = \Omega(t\gamma + p^{-t})$. For $t = O(\log 1/\mu)$, we get $\gamma \geq \Omega(\mu/(\log 1/\mu))$.

In the standard way, the sequence of implications above yields an efficient randomized algorithm, with the runtime $\mathsf{poly}(k, m, \log 1/\mu)$, for going in the reverse direction: from a predictor circuit for h^{vN} to a predictor circuit for g^{GL}. To be able to carry out the hybrid argument with uniform algorithms, we need efficient sampleability of the distribution $(\mathcal{D}, g^{GL}(\mathcal{D}))$. Such sampling is possible when we have membership queries to f (as $g^{GL} \in \mathsf{P}^f$); in fact, here it would suffice to have access to uniformly random labeled examples $(x, f(x))$. Another issue is that we need to sample uniformly from $\mathbb{Z}_p$, while we only have access to uniformly random bits. However, it is easy to devise an efficient sampling algorithm for $\mathbb{Z}_p$, with the distribution statistically almost indistinguishable from uniform over $\mathbb{Z}_p$.[5] ◀

We now have all the ingredients to prove the following.

▶ **Theorem 4.8** (Black-Box Amplification within $\mathsf{AC}^0[p]$)**.** *For any $0 < \epsilon, \gamma < 1$, there is black-box (ϵ,γ)-amplification within $\mathsf{AC}^0[p]$.*

Proof. The proof is similar to that of Theorem 4.3. To amplify a given function f, we first apply the Direct Product construction to get $g = f^k$ (for an appropriate parameter k), then use the Goldreich-Levin construction to get g^{GL}, and finally apply the von Neumann construction h^{vN}. The only difference is the use of the von Neumann construction of Theorem 4.7. But the only consequence of this extra step for the parameters of the amplification procedure is the slightly worse dependence on $1/\gamma$: from $1/\gamma$ to $(1/\gamma)\cdot\log 1/\gamma \leq 1/\gamma^2$. ◀

5 Natural properties imply randomized learning

In this section, we prove the general implication from natural properties to learning algorithms. First we prove the generic reduction from learning (and compression) to natural properties. Then, as our main application, we use the known natural properties for $\mathsf{AC}^0[p]$, to get learning and compression algorithms for $\mathsf{AC}^0[p]$.

[5] We divide an interval $[0, 2^{k-1}]$ into p almost equal pieces (all but the last piece are equal to $\lfloor 2^k/p \rfloor$), and check in AC^0 which piece we fall into. The statistical difference between the uniform distribution over $\mathbb{Z}_p$ and this distribution is at most $p/2^k$. So we can make it negligible by choosing k to be a large enough polynomial in the relevant parameters.

5.1 A generic reduction from learning to natural properties

▶ **Theorem 5.1** (Learning from a natural property). *Let Λ be any circuit class containing $\mathsf{AC}^0[p]$ for some prime p. Let R be a P-natural property, with largeness at least $1/5$, that is useful against $\Lambda[u]$, for some size function $u\colon \mathbb{N} \to \mathbb{N}$. Then there is a randomized algorithm that, given oracle access to any function $f\colon \{0,1\}^n \to \{0,1\}$ from $\Lambda[s_f]$, produces a circuit $C \in \widetilde{\mathsf{CKT}}(f,\epsilon)$ in time $\mathsf{poly}(n, 1/\epsilon, 2^{u^{-1}(\mathsf{poly}(n,1/\epsilon,s_f))})$.*

Proof. Let $\textsc{Gen}(f) = \{g_z\}$ be an (ϵ, L)-black-box generator based on f, for $L(n)$ such that $\log L(n) > u^{-1}(\mathsf{poly}(n, 1/\epsilon(n), s_f))$. Using the nonuniform Λ-efficiency of black-box generators, we have that $g_z \in \Lambda^f[\mathsf{poly}(n, 1/\epsilon)]$, for every z. Hence, we get, by replacing the f-oracle with the Λ-circuit for f, that $g_z \in \Lambda[s_g]$, for some $s_g \leq \mathsf{poly}(n, 1/\epsilon, s_f)$. We want $s_g < u(\log L(n))$. This is equivalent to $u^{-1}(s_g) < \log L(n)$.

Let D be the circuit obtained from the natural property R restricted to truth tables of size $L(n)$. By usefulness, we have $\Pr_z[\neg D(g_z) = 1] = 1$, and by largeness, $\Pr_y[\neg D(y) = 1] \leq 1 - 1/5$. So $\neg D$ is a $1/5$-distinguisher for $\textsc{Gen}(f)$. By the reconstruction property of black-box generators, we have a randomized algorithm that constructs a circuit $C \in \widetilde{\mathsf{CKT}}(f,\epsilon)$ in time $\mathsf{poly}(n, 1/\epsilon(n), L(n)) = \mathsf{poly}(n, 1/\epsilon, 2^{u^{-1}(\mathsf{poly}(n,1/\epsilon,s_f))})$, as required. ◀

For different usefulness bounds u, we get different runtimes for our learning algorithm:

- polynomial $\mathsf{poly}(ns_f/\epsilon)$, for $u(n) = 2^{\Omega(n)}$,
- quasi-polynomial $\mathsf{quasi\text{-}poly}(ns_f/\epsilon)$, for $u(n) = 2^{n^\alpha}$ where $\alpha < 1$, and
- subexponential $\mathsf{poly}(n, 1/\epsilon, 2^{(ns_f/\epsilon)^{o(1)}})$, for $u(n) = n^{\omega(1)}$.

▶ **Corollary 5.2.** *Under the same assumptions as in Theorem 5.1, we also get randomized compression for $\Lambda[\mathsf{poly}]$ to the circuit size at most $O(\epsilon(n) \cdot 2^n \cdot n)$, for any $0 < \epsilon(n) < 1$ such that $\log(\epsilon(n) \cdot 2^n \cdot n) \geq u^{-1}(\mathsf{poly}(n, 1/\epsilon))$.*

Proof. We use Theorem 5.1 to learn a small circuit that computes f on all but at most $\epsilon \cdot 2^n$ inputs, and then patch up this circuit by hardwiring all the error inputs, using extra circuitry of size at most $O(\epsilon \cdot 2^n \cdot n)$. This size will dominate the overall size of the patched-up circuit for the ϵ satisfying the stated condition. ◀

5.2 Application: Learning and compression algorithms for $\mathsf{AC}^0[p]$

We have natural properties useful against the class of AC^0 circuits with mod p gates, for any fixed prime p, as given in [31]. The lower bound of Razborov [30] (showing that $\textsc{Majority}$ is not in $\mathsf{AC}^0[2]$) embeds a natural property against $\mathsf{AC}^0[2]$, and the lower bound of Smolensky [35] (showing that $\textsc{Parity}$ is not in $\mathsf{AC}^0[p]$, for any prime $p \neq 2$) embeds a natural property against $\mathsf{AC}^0[p]$ for any prime $p > 2$. In both cases, the natural property is NC^2-computable, and is useful for circuit size up to $2^{\Omega(n^{1/(2d)})}$, where d is the circuit depth, and n is the input size.

▶ **Theorem 5.3** ([31]). *For every prime p, there is an NC^2-natural property of n-variate Boolean functions, with largeness at least $1/2$, that is useful against $\mathsf{AC}^0[p]$ circuits of depth d of size up to $2^{\Omega(n^{1/(2d)})}$.*

Below we sketch the corresponding natural properties; see the full paper for more details.

Natural Property useful against $\mathsf{AC}^0[2]$. For $0 \le a, b \le n$, define a linear transformation $A_{a,b}$ that maps a Boolean function $f\colon \{0,1\}^n \to \{0,1\}$ to a matrix $M = A_{a,b}(f)$ of dimension $\binom{n}{a} \times \binom{n}{b}$, whose rows are indexed by size a subsets of $[n]$, and rows by size b subsets of $[n]$. For every $K \subseteq [n]$, define the set $Z(K) = \{(x_1, \ldots, x_n) \in \{0,1\}^n \mid \forall i \in K,\ x_i = 0\}$. For a size a subset $I \subseteq [n]$ and size b subset $J \subseteq [n]$, define $M_{I,J} = \oplus_{x \in Z(I \cup J)} f(x)$.

The natural property of Theorem 5.3 for $\mathsf{AC}^0[2]$ is the following algorithm:

> Given an n-variate Boolean function f, construct matrices $M_b = A_{a,b}(f)$ for $a = n/2 - \sqrt{n}$ and for every $0 \le b \le a$. Accept f if, for at least one b, $\mathrm{rank}(M_b) \ge \frac{2^n}{140n^2}$.

Natural Property useful against $\mathsf{AC}^0[p]$, for primes $p > 2$. Let f be a given n-variate Boolean function. Without loss of generality, assume n is odd. Denote by L the vector space of all multilinear polynomials of degree less than $n/2$ over $\mathbf{GF}(p)$. Let $\bar{f}$ be the unique multilinear polynomial over $\mathbf{GF}(p)$ that represents f on the Boolean cube $\{-1,1\}^n$ (after the linear transformation mapping the Boolean 0 to $1 \mod p$, and the Boolean 1 to $-1 \mod p$), i.e., f and $\bar{f}$ agree over all points of $\{-1,1\}^n$.

The natural property of Theorem 5.3 for $\mathsf{AC}^0[p]$ is the following algorithm:

> Given an n-variate Boolean function f, construct its unique multilinear polynomial extension $\bar{f}$ over $\mathbf{GF}(p)$. Accept f if $\dim(\bar{f}L + L) \ge \frac{3}{4} \cdot 2^n$ (over $\mathbf{GF}(p)$).

Theorem 5.3, in conjunction with Theorem 5.1, immediately yields our main application.

▶ **Corollary 5.4** (Learning $\mathsf{AC}^0[p]$ in quasipolytime)**.** *For every prime p, there is a randomized algorithm that, using membership queries, learns a given n-variate Boolean function $f \in \mathsf{AC}^0[p]$ of size s_f to within error ϵ over the uniform distribution, in time* $\mathsf{quasi\text{-}poly}(ns_f/\epsilon)$.

Using Corollary 5.2, we also immediately get the following compression result, first proved (with somewhat stronger parameters) by Srinivasan [36].

▶ **Corollary 5.5.** *There is a randomized compression algorithm for depth-d $\mathsf{AC}^0[p]$ functions that compresses an n-variate function to the circuit size at most $2^{n-n^{\mu}}$, for $\mu \ge \Omega(1/d)$.*

5.3 Sketch of Complete Algorithm

Here, we sketch the algorithm implied by Theorem 5.1 for the case of $\mathsf{AC}^0[2]$. Let $f\colon \{0,1\}^n \to \{0,1\}$ be a function in $\mathsf{AC}^0[2]$ to be learned, given via membership oracle. Let R be a natural property, and let $L = n^{\mathsf{poly}(\log n)}$.

1. Design a subroutine for computing $\textsc{Amp}(f) = (f^k)^{GL}$ (Theorem 4.3) using f as an oracle.
2. Let D be a circuit simulating the natural property R_L. D is a distinguisher between $G_{\textsc{Amp}(f)}(s)$ for a random s and uniform, as shown in the proof of Theorem 5.1.
3. Convert D into C, a weak predictor for $\textsc{Amp}(f)$ on $(1/2 + \Omega(1/L))$-fraction of inputs, using the NW reconstruction algorithm (Section 2.9) and oracle for $\textsc{Amp}(f)$.
4. Use C as the oracle for the Goldreich-Levin reconstruction algorithm (Theorem 4.2), obtaining a predictor C' for the direct product of f.
5. Use C' as input to the Direct Product reconstruction algorithm of Theorem 4.1, and print the resulting circuit.

For the case of $\mathsf{AC}^0[p]$ with $p \neq 2$, the algorithm is essentially the same, but requires an additional step in the definition of $\textsc{Amp}(f)$: the von Neumann construction (Theorem 4.7) applied to $(f^k)^{GL}$. Thus, we need the von Neumann reconstruction step inserted between steps 3 and 4 of the complete algorithm above.

6 NW designs cannot be computed in AC^0

In Section 3.1 we showed that NW designs (with parameters of interest to us) are computable by small $\mathsf{AC}^0[p]$ circuits, for any prime p. It is natural to ask if one can compute such NW designs by small AC^0 circuits, without modular gates. Here we show that this is *not* possible.

Consider an NW design $S_1, \ldots, S_L \subseteq [n^2]$, where

- each set S_i is of size n,
- the number of sets is $L = 2^\ell$ for $\ell = n^\epsilon$ (for some $\epsilon > 0$), and
- for any two distinct sets S_i and S_j, $i \neq j$, we have $|S_i \cap S_j| \leq \ell$.

To describe such a design, we use the following Boolean function g: for $1 \leq i \leq L$, and for $1 \leq k \leq n^2$, define $g(i,k) = 1$ iff $k \in S_i$.

We will prove the following.

▶ **Theorem 6.1.** *Let $g\colon \{0,1\}^{\ell+2\log n} \to \{0,1\}$ be the characteristic function for any NW design with the above parameters. Then g requires depth d AC^0 circuits of size $\exp(\ell^{1/d})$.*

To prove this result, we shall define a family of Boolean functions f_T, parameterized by sets $T \subseteq [n^2]$: for $1 \leq i \leq L$, we define $f_T(i) = 1$ iff $T \cap S_i \neq \emptyset$. Observe that if $g(i,k)$ is computable by AC^0 circuits of depth d and size s, then, for every set T, the function $f_T(i) = \vee_{k \in T}\ g(i,k)$ is computable by AC^0 circuits of depth at most $d+1$ and size $O(s \cdot |T|)$. Therefore, to prove Theorem 6.1, it will suffice to prove the following.

▶ **Lemma 6.2.** *There exists a set $T \subseteq [n^2]$ such that $f_T\colon \{0,1\}^\ell \to \{0,1\}$ requires depth $d+1$ AC^0 circuits of size at least $\exp(\ell^{1/d})$.*

The idea of the proof of Lemma 6.2 is to show that for a random set T (of expected size $O(n)$), the function f_T has high average sensitivity (i.e., is likely to flip its value for many Hamming neighbors of a randomly chosen input). By averaging, we get the existence of a particular function f_T of high average sensitivity. On the other hand, it is well-known that AC^0 functions have low average sensitivity. This will imply that f_T must require large AC^0 circuits. We refer the reader to the full version of the paper for more details.

7 Conclusions

For our applications, we need Λ strong enough to carry out a (version of) the construction, yet weak enough to have a natural property useful against it. Here we show that $\Lambda = \mathsf{AC}^0[p]$ for any prime p satisfies both conditions. A logical next step would be ACC^0: if one can get a natural property useful against ACC^0, for example by naturalizing Williams's [43] proof, then a learning algorithm for ACC^0 would follow. (As MOD_p can be simulated with MOD_m, $m = p \cdot a$ gates by duplicating each input to the Mod_m gate a times (without any penalty in the number of gates), our construction for MOD_p can be applied directly by taking p to be any prime factor of m.)

Connections between learning algorithms and lower bounds could also be explored in other settings. In particular, it would be interesting to give such a connection for arithmetic circuits. In [23], the NW generator is used to derandomize polynomial identity testing based on a polynomial with a large arithmetic circuit lower bound. Since the main reduction is constructive, one might hope to use it to design learning (or interpolation) algorithms for multivariate polynomials of small circuit complexity. However, it is unclear what the analogy of "natural property" would be in this setting.

We conclude with the following open questions. Can we get an *exact* compression algorithm for $\mathsf{AC}^0[p]$ (or even AC^0) functions that would produce circuits of *subexponential*

size? Can our learning algorithm be derandomized? Is there a way to get nontrivial SAT algorithms from natural properties? Finally, are there more applications of "play-to-lose" pseudorandom constructions?

Acknowledgments. This work was partially supported by the Simons Foundation and NSF grants #CNS-1523467 and CCF-121351 (M. Carmosino, R. Impagliazzo) and by NSERC Discovery grants (V. Kabanets, A. Kolokolova). This work was done in part while all authors were visiting Simons Institute for the Theory of Computing. We thank the anonymous referees for their helpful suggestions.

References

1 Baris Aydinlioglu and Dieter van Melkebeek. Nondeterministic circuit lower bounds from mildly de-randomizing Arthur-Merlin games. In *Proceedings of the 27th Conference on Computational Complexity, CCC 2012, Porto, Portugal, June 26-29, 2012*, pages 269–279, 2012.

2 László Babai, Lance Fortnow, Noam Nisan, and Avi Wigderson. BPP has subexponential time simulations unless EXPTIME has publishable proofs. *Computational Complexity*, 3:307–318, 1993. doi:10.1007/BF01275486.

3 Paul Beame, Russell Impagliazzo, and Srikanth Srinivasan. Approximating AC^0 by small height decision trees and a deterministic algorithm for $\#AC^0$SAT. In *Proceedings of the 27th Conference on Computational Complexity, CCC 2012, Porto, Portugal, June 26-29, 2012*, pages 117–125, 2012.

4 Mark Braverman. Polylogarithmic independence fools AC^0 circuits. *Journal of the ACM*, 57:28:1–28:10, 2010.

5 Marco Carmosino, Russell Impagliazzo, Valentine Kabanets, and Antonina Kolokolova. Tighter connections between derandomization and circuit lower bounds. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX-/RANDOM 2015, August 24-26, 2015, Princeton, NJ, USA*, pages 645–658, 2015.

6 Ruiwen Chen and Valentine Kabanets. Correlation bounds and #SAT algorithms for small linear-size circuits. In Dachuan Xu, Donglei Du, and Dingzhu Du, editors, *Computing and Combinatorics – 21st International Conference, COCOON 2015, Beijing, China, August 4-6, 2015, Proceedings*, volume 9198 of *Lecture Notes in Computer Science*, pages 211–222. Springer, 2015. doi:10.1007/978-3-319-21398-9_17.

7 Ruiwen Chen, Valentine Kabanets, Antonina Kolokolova, Ronen Shaltiel, and David Zuckerman. Mining circuit lower bound proofs for meta-algorithms. *Computational Complexity*, 24(2):333–392, 2015. doi:10.1007/s00037-015-0100-0.

8 Ruiwen Chen, Valentine Kabanets, and Nitin Saurabh. An improved deterministic #SAT algorithm for small de Morgan formulas. In *Mathematical Foundations of Computer Science 2014 – 39th International Symposium, MFCS 2014, Budapest, Hungary, August 25-29, 2014. Proceedings, Part II*, pages 165–176, 2014.

9 Ruiwen Chen and Rahul Santhanam. Improved algorithms for sparse MAX-SAT and MAX-k-CSP. In *Theory and Applications of Satisfiability Testing – SAT 2015 – 18th International Conference, Austin, TX, USA, September 24-27, 2015, Proceedings*, pages 33–45, 2015. doi:10.1007/978-3-319-24318-4_4.

10 Lance Fortnow and Adam R. Klivans. Efficient learning algorithms yield circuit lower bounds. *J. Comput. Syst. Sci.*, 75(1):27–36, 2009. doi:10.1016/j.jcss.2008.07.006.

11 Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *J. ACM*, 33(4):792–807, 1986.

12 Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In David S. Johnson, editor, *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washigton, USA*, pages 25–32. ACM, 1989. doi:10.1145/73007.73010.

13 Oded Goldreich, Noam Nisan, and Avi Wigderson. On Yao's XOR-lemma. In *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation*, pages 273–301. Springer, 2011.

14 Oded Goldreich, Ronitt Rubinfeld, and Madhu Sudan. Learning polynomials with queries: The highly noisy case. *SIAM J. Discrete Math.*, 13(4):535–570, 2000. doi:10.1137/S0895480198344540.

15 Johan Håstad. Almost optimal lower bounds for small depth circuits. In S. Micali, editor, *Randomness and Computation*, pages 143–170, Greenwich, Connecticut, 1989. Advances in Computing Research, vol. 5, JAI Press.

16 Alexander Healy, Salil Vadhan, and Emanuele Viola. Using nondeterminism to amplify hardness. *SIAM Journal on Computing*, 35(4):903–931, 2006.

17 Russell Impagliazzo, Ragesh Jaiswal, Valentine Kabanets, and Avi Wigderson. Uniform direct product theorems: Simplified, optimized, and derandomized. *SIAM J. Comput.*, 39(4):1637–1665, 2010. doi:10.1137/080734030.

18 Russell Impagliazzo, Valentine Kabanets, and Avi Wigderson. In search of an easy witness: Exponential time vs. probabilistic polynomial time. *J. Comput. Syst. Sci.*, 65(4):672–694, 2002. doi:10.1016/S0022-0000(02)00024-7.

19 Russell Impagliazzo, William Matthews, and Ramamohan Paturi. A satisfiability algorithm for AC^0. In Yuval Rabani, editor, *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan, January 17-19, 2012*, pages 961–972. SIAM, 2012. doi:10.1137/1.9781611973099.

20 Russell Impagliazzo, Raghu Meka, and David Zuckerman. Pseudorandomness from shrinkage. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 111–119, 2012. doi:10.1109/FOCS.2012.78.

21 Russell Impagliazzo and Avi Wigderson. $P = BPP$ if E requires exponential circuits: Derandomizing the XOR lemma. In Frank Thomson Leighton and Peter W. Shor, editors, *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing, El Paso, Texas, USA, May 4-6, 1997*, pages 220–229. ACM, 1997. doi:10.1145/258533.258590.

22 Russell Impagliazzo and Avi Wigderson. Randomness vs time: Derandomization under a uniform assumption. *J. Comput. Syst. Sci.*, 63(4):672–688, 2001. doi:10.1006/jcss.2001.1780.

23 Valentine Kabanets and Russell Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Computational Complexity*, 13(1-2):1–46, 2004. doi:10.1007/s00037-004-0182-6.

24 Adam Klivans, Pravesh Kothari, and Igor Carboni Oliveira. Constructing hard functions using learning algorithms. In *Proceedings of the 28th Conference on Computational Complexity, CCC 2013, Palo Alto, California, USA, 5-7 June, 2013*, pages 86–97. IEEE, 2013. doi:10.1109/CCC.2013.18.

25 Nathan Linial, Yishay Mansour, and Noam Nisan. Constant depth circuits, Fourier transform, and learnability. *J. ACM*, 40(3):607–620, 1993. doi:10.1145/174130.174138.

26 Oleg B. Lupanov. On the synthesis of switching circuits. *Soviet Mathematics*, 119(1):23–26, 1958. English translation in *Soviet Mathematics Doklady*.

27 Oleg B. Lupanov. A method of circuit synthesis. *Izvestiya VUZ, Radiofizika*, 1(1):120–140, 1959. (in Russian).

28 Noam Nisan and Avi Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994. doi:10.1016/S0022-0000(05)80043-1.

29 Ryan O'Donnell. Hardness amplification within NP. *J. Comput. Syst. Sci.*, 69(1):68–94, 2004.

30 Alexander A. Razborov. Lower bounds on the size of bounded depth circuits over a complete basis with logical addition. *Mathematical Notes*, 41(4):333–338, 1987.

31 Alexander A. Razborov and Steven Rudich. Natural proofs. *J. Comput. Syst. Sci.*, 55(1):24–35, 1997. doi:10.1006/jcss.1997.1494.

32 Rahul Santhanam. Fighting perebor: New and improved algorithms for formula and QBF satisfiability. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*, pages 183–192. IEEE Computer Society, 2010. doi:10.1109/FOCS.2010.25.

33 Rahul Santhanam and Richard Ryan Williams. Beating exhaustive search for quantified boolean formulas and connections to circuit complexity. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 231–241, 2015. doi:10.1137/1.9781611973730.18.

34 Kazuhisa Seto and Suguru Tamaki. A satisfiability algorithm and average-case hardness for formulas over the full binary basis. In *Proceedings of the Twenty-Seventh Annual IEEE Conference on Computational Complexity*, pages 107–116, 2012.

35 Roman Smolensky. Algebraic methods in the theory of lower bounds for boolean circuit complexity. In *Proceedings of the nineteenth annual ACM symposium on Theory of computing*, pages 77–82. ACM, 1987.

36 Srikanth Srinivasan. A compression algorithm for $AC^0[\oplus]$ circuits using certifying polynomials. *Electronic Colloquium on Computational Complexity (ECCC)*, 22:142, 2015. URL: http://eccc.hpi-web.de/report/2015/142.

37 Avishay Tal. #SAT algorithms from shrinkage. *Electronic Colloquium on Computational Complexity (ECCC)*, 22:114, 2015. URL: http://eccc.hpi-web.de/report/2015/114.

38 Luca Trevisan. On uniform amplification of hardness in NP. In *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 31–38. ACM, 2005.

39 Christopher Umans. Pseudo-random generators for all hardnesses. *J. Comput. Syst. Sci.*, 67(2):419–440, 2003. doi:10.1016/S0022-0000(03)00046-1.

40 John von Neumann. Various techniques used in connection with random digits. *J. Research Nat. Bur. Stand., Appl. Math. Series*, 12:36–38, 1951.

41 Ryan Williams. Improving exhaustive search implies superpolynomial lower bounds. *SIAM J. Comput.*, 42(3):1218–1244, 2013. doi:10.1137/10080703X.

42 Ryan Williams. New algorithms and lower bounds for circuits with linear threshold gates. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 – June 03, 2014*, pages 194–202, 2014. doi:10.1145/2591796.2591858.

43 Ryan Williams. Nonuniform ACC circuit lower bounds. *J. ACM*, 61(1):2:1–2:32, 2014. doi:10.1145/2559903.

Decoding Reed-Muller Codes Over Product Sets*

John Y. Kim[1] and Swastik Kopparty[2]

1 **Department of Mathematics, Rutgers University, Piscataway, USA**
jonykim@math.rutgers.edu

2 **Department of Mathematics & Department of Computer Science, Rutgers University, Piscataway, USA**
swastik@math.rutgers.edu

Abstract

We give a polynomial time algorithm to decode multivariate polynomial codes of degree d up to half their minimum distance, when the evaluation points are an arbitrary product set S^m, for every $d < |S|$. Previously known algorithms can achieve this only if the set S has some very special algebraic structure, or if the degree d is significantly smaller than $|S|$. We also give a near-linear time algorithm, which is based on tools from list-decoding, to decode these codes from nearly half their minimum distance, provided $d < (1-\epsilon)|S|$ for constant $\epsilon > 0$.

Our result gives an m-dimensional generalization of the well known decoding algorithms for Reed-Solomon codes, and can be viewed as giving an algorithmic version of the Schwartz-Zippel lemma.

1998 ACM Subject Classification E.4 Coding and Information Theory

Keywords and phrases polynomial codes, Reed-Muller codes, coding theory, error-correcting codes

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.11

1 Introduction

Error-correcting codes based on polynomials have played an important role throughout the history of coding theory. The mathematical phenomenon underlying these codes is that distinct low-degree polynomials have different evaluations at many points. More recently, the intimate relation between polynomials and computation has led to polynomial-based error-correcting codes having a big impact on complexity theory. Notable applications include PCPs, interactive proofs, polynomial identity testing and property testing.

Our main result is a decoding algorithm for multivariate polynomial codes. Let $\mathbb{F}$ be a field, let $S \subseteq \mathbb{F}$, let $d < |S|$ and let $m \geq 1$. Consider the code of all m-variate polynomials of total degree at most d, evaluated at all points of S^m:

$$\mathcal{C} = \{\langle P(\mathbf{a})\rangle_{\mathbf{a}\in S^m} \mid P(X_1,\ldots,X_m) \in \mathbb{F}[X_1,\ldots,X_m], \deg(P) \leq d\}.$$

When $m = 1$, this code is known as the Reed-Solomon code [3], and for $m > 1$ this code is known as the Reed-Muller code [1, 2][1].

* Research supported in part by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-1433187, NSF grant CCF-1253886, and a Sloan Fellowship.

[1] The family of Reed-Muller codes also includes polynomial evaluation codes where the individual degree d is larger than $|S|$, and the individual degree is capped to be at most $|S|-1$. We do not consider the $d \geq |S|$ case in this paper.

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 11; pp. 11:1–11:28

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

The code $\mathcal{C}$ above is a subset of $\mathbb{F}^{S^m}$, which we view as the space of functions from S^m to $\mathbb{F}_q$. Given two functions $f, g : S^m \to \mathbb{F}$, we define their (relative Hamming) distance $\Delta(f, g) = \Pr_{\mathbf{a}\in S^m}[f(\mathbf{a}) \neq g(\mathbf{a})]$. To understand the error-correcting properties of $\mathcal{C}$, we recall the following well known lemma, often called the Schwartz-Zippel lemma:

▶ **Lemma 1.1.** *Let $\mathbb{F}$ be a field, and let $P(X_1, \ldots, X_m)$ be a nonzero polynomial over $\mathbb{F}$ with degree at most d. Then for every $S \subseteq \mathbb{F}$,*

$$\Pr_{\mathbf{a}\in S^m}[P(\mathbf{a}) = 0] \leq \frac{d}{|S|}.$$

This lemma implies that for any two polynomials P, Q of degree at most d, $\Delta(P, Q) \geq (1 - \frac{d}{|S|})$. In other words the minimum distance of $\mathcal{C}$ is at least $(1 - \frac{d}{|S|})$. It turns out that the minimum distance of $\mathcal{C}$ is in fact exactly $(1 - \frac{d}{|S|})$, and we let $\delta_{\mathcal{C}}$ denote this quantity.

For error-correcting purposes, if we are given a "received word" $r : S^m \to \mathbb{F}$ such that there exists a polynomial P of degree at most d with $\Delta(r, P) \leq \delta_{\mathcal{C}}/2$, then we know that there is a unique such P. The problem that we consider in this paper, "decoding $\mathcal{C}$ upto half its minimum distance", is the algorithmic task of finding this P.

1.1 Our Results

There is a rich history with several deep algebraic ideas surrounding the problem of decoding multivariate polynomial codes. We first state our main results, and then discuss its relationship to the various other known results.

▶ **Theorem 1.2** (Efficient decoding of multivariate polynomial codes upto half their minimum distance)**.** *Let $\mathbb{F}$ be a finite field, let S, d, m be as above, and let $\delta_{\mathcal{C}} = (1 - \frac{d}{|S|})$.*

There is an algorithm, which when given as input a function $r : S^m \to \mathbb{F}$, runs in time $\mathsf{poly}(|S|^m, \log|\mathbb{F}|)$ finds the polynomial $P(X_1, \ldots, X_m) \in \mathbb{F}[X_1, \ldots, X_m]$ of degree at most d (if any) such that:

$$\Delta(r, P) < \delta_{\mathcal{C}}/2.$$

As we will discuss below, previously known efficient decoding algorithms for these codes only either worked for (1) very algebraically special sets S, or (2) very low degrees d, or (3) decoded from a much smaller fraction of errors ($\approx \frac{1}{m+1}\delta_{\mathcal{C}}$ instead of $\frac{1}{2}\delta_C$).

Using several further ideas, we also show how to implement the above algorithm in near-linear time to decode upto almost half the minimum distance, provided d is not $(1 - o(1))|S|$.

▶ **Theorem 1.3** (Near-linear time decoding)**.** *Let $\mathbb{F}$ be a finite field, let S, d, m be as above, and let $\delta_{\mathcal{C}} = (1 - \frac{d}{|S|})$. Assume $\delta_C > 0$ is a constant.*

There is an algorithm, which when given as input a function $r : S^m \to \mathbb{F}$, runs in time $|S|^m \cdot \mathsf{poly}(\log|S|^m, \log|\mathbb{F}|)$ finds the polynomial $P(X_1, \ldots, X_m) \in \mathbb{F}[X_1, \ldots, X_m]$ of degree at most d (if any) with:

$$\Delta(r, P) < (1 - o(1)) \cdot \delta_{\mathcal{C}}/2.$$

Over the rational numbers, we get a version of Theorem 1.2 where the running time is $\mathsf{poly}(|S|^m, t)$, where t is the maximum bit-complexity of any point in S or in the image of r. This enables us to decode multivariate polynomial codes upto half the minimum distance in the natural special case where the evaluation set S equals $\{1, 2, \ldots, n\}$.

We also mention that decoding Reed-Muller codes over an arbitrary product set S^m appears as a subroutine in the local decoding algorithm for multiplicity codes [17] (see Section 4 on "Solving the noisy system"). Our results allow the local decoding algorithms there to run efficiently over all fields ([17] could only do this over fields of small characteristic, where algebraically special sets S are available).

1.2 Related work

There have been many works studying the decoding of multivariate polynomial codes, which prove (and improve) various special cases of our main theorem.

1.2.1 Reed-Solomon codes ($m = 1$)

When $m = 1$, our problem is also known as the problem of decoding Reed-Solomon codes upto half their minimum distance. That this problem can be solved efficiently is very classical, and a number of algorithms are known for this (Mattson-Solomon [5], Berlekamp-Massey [4], Berlekamp-Welch [12]). The underlying algorithmic ideas have subsequently had a tremendous impact on algebraic algorithms.

For Reed-Solomon codes, it is in fact known how to list-decode beyond half the minimum distance, upto the Johnson bound (Guruswami-Sudan [6]). This has had numerous further applications in coding theory, complexity theory and pseudorandomness.

1.2.2 Special sets S

For very special sets S, it turns out that there are some algebraic ways to reduce the decoding of multivariate polynomial codes over S^m to the decoding of univariate polynomial codes. This kind of reduction is possible when S equals the whole field $\mathbb{F}$, or more generally when S equals an affine subspace over the prime subfield of $\mathbb{F}$.

When $S = \mathbb{F}_q$, then $S^m = \mathbb{F}_q^m$ and S^m can then be identified with the large field $\mathbb{F}_{q^m}$ in a natural $\mathbb{F}_q$-linear way (this understanding of Reed-Muller codes was discovered by [8]). This converts the multivariate setting into univariate setting, identifies the multivariate polynomial code as a subcode of the univariate polynomial code, and (somewhat miraculously), the minimum distance of the univariate polynomial code equals the minimum distance of the multivariate polynomial code. Thus the classical Reed-Solomon decoding algorithms can then be used, and this leads to an algorithm for the multivariate setting decoding upto half the minimum distance. In fact, Pellikaan-Wu [7] observed that this connection allows one to decode multivariate polynomial codes beyond half the minimum distance too, provided S is special in the above sense.

Another approach which works in the case of $S = \mathbb{F}_q$ is based on local decoding. Here we use the fact that $S^m = \mathbb{F}_q^m$ contains many lines (not just the axis-parallel ones), and then use the univariate decoding algorithms to decode on those lines from $(1 - \frac{d}{q})/2$ fraction errors. This approach manages to decode multivariate polynomial codes with $S = \mathbb{F}_q$ from $(\frac{1}{2} - o(1))$ of the minimum distance. Again, this approach does not work for general S, since a general S^m usually contains only axis-parallel lines (while $\mathbb{F}_q^m$ has many more lines).

1.2.3 Low degree d

When the degree d of the multivariate polynomial code is significantly smaller than $|S|$, then a number of other list-decoding based methods come into play.

The powerful Reed-Muller list-decoding algorithm of Sudan [9] and its multiplicity-based generalization, based on $(m+1)$-variate interpolation and root-finding, can decode from $1-(\frac{d}{|S|})^{\frac{1}{m+1}}$ fraction errors. With small degree $d = o(|S|)$ and $m = O(1)$, this decoding radius equals $1 - o(1)$! However when d is much larger (say $0.9 \cdot |S|$), then the fraction of errors decodable by this algorithm is around $\frac{1}{m+1} \cdot (1 - \frac{d}{|S|}) = \frac{1}{m+1} \cdot \delta_C$.

Another approach comes from the list-decoding of tensor codes [10]. While the multivariate polynomial codes we are interested in are not tensor codes, they are subcodes of the code of polynomials with *individual degree* at most d. Using the algorithm of [10] for decoding tensor codes, we get an algorithm that can decode from a $1-o(1)$ fraction of errors when $d = o(|S|)$, but fails to approach a constant fraction of the minimum distance when d approaches $|S|$.

In light of all the above, to the best of our knowledge, for multivariate polynomial codes with $d > 0.9 \cdot |S|$ (i.e., $\delta_C < 0.1$), and S generic, the largest fraction of errors which could be corrected efficiently was about $\frac{1}{m+1}\delta_C$. In particular, the correctable fraction of errors is a vanishing fraction of the minimum distance, as the number of variables m grows.

We thus believe it is worthwhile to investigate this problem, not only because of its basic nature, but also because of the many different powerful algebraic ideas that only give partial results towards it.

1.3 Overview of the decoding algorithm

We now give a brief overview of our decoding algorithms. Let us first discuss the bivariate ($m = 2$) case. Here we are given a received word $r : S^2 \to \mathbb{F}$ such that there exists a codeword $P(X, Y) \in \mathbb{F}[X, Y]$ of degree at most $d = (1 - \delta_C)|S|$ with $\Delta(P, r) < \frac{\delta_C}{2}$. Our goal is to find $P(X, Y)$.

First some high-level strategy. An important role in our algorithm is played by the following observation: the restriction of a degree $\leq d$ bivariate polynomial $P(X, Y)$ to a vertical line (fixing $X = \alpha$) or a horizontal line (fixing $Y = \beta$) gives a degree $\leq d$ univariate polynomial. Perhaps an even more important role is played by the following disclaimer: *the previous observation does not characterize bivariate polynomials of degree d!* The set of functions $f : S^2 \to \mathbb{F}$ for which the horizontal restrictions and vertical restrictions are polynomials of degree $\leq d$ is the code of polynomials with *individual degree* at most d (this is the tensor Reed-Solomon code, with much smaller distance than the Reed-Muller code). For such a function f to be in the Reed-Muller code, the different univariate polynomials that appear as horizontal and vertical restrictions must be related in some way. The crux of our algorithm is to exploit these relations.

It will also help to recap the standard algorithm to decode tensor Reed-Solomon codes upto half their minimum distance (this scheme actually works for general tensor codes). Suppose we are given a received word $r : S^2 \to \mathbb{F}$, and we want to find a polynomial $P(X, Y)$ with individual degrees at most d which is close to r. One then takes the rows of this new received word (after having corrected the columns) and decodes them to the nearest degree $\leq d$ polynomial. The key point is to pass some "soft information" from the column decodings to the row decodings; the columns which were decoded from more errors are treated with lower confidence. This decodes the tensor Reed-Solomon code from $1/2$ the minimum distance fraction errors. Several ingredients from this algorithm will appear in our Reed-Muller decoding algorithm.

Now we return to the problem of decoding Reed-Muller codes. Let us write $P(X, Y)$ as a single variable polynomial in Y with coefficients in $\mathbb{F}[X]$: $P(X, Y) = \sum_{i=0}^{d} P_i(X)Y^{d-i}$, where $\deg(P_i) \leq i$. For each $\alpha \in S$, consider the restricted univariate polynomial $P(\alpha, Y)$. Since $\deg(P_0) = 0$, $P_0(\alpha)$ must be the same for each α. Thus all the polynomials $\langle P(\alpha, Y)\rangle_{\alpha \in S}$

have the same coefficient for Y^d. Similarly, the coefficients of Y^{d-i} in the polynomials $\langle P(\alpha, Y)\rangle_{\alpha\in S}$ fit a degree i polynomial.

As in the tensor Reed-Solomon case, our algorithm begins by decoding each column $r(\alpha, \cdot)$ to the nearest degree $\leq d$ univariate polynomial. Now, instead of trying to use these decoded column polynomials to recover $P(X, Y)$ in one shot, we aim lower and just try to recover $P_0(X)$. The advantage is that $P_0(X)$ is only a degree 0 polynomial, and is thus resilient to many more errors than a degree d polynomial. Armed with $P_0(X)$, we then proceed to find $P_1(X)$. The knowledge of $P_0(X)$ allows us to decode the columns $r(\alpha, \cdot)$ to a slightly larger radius; in turn this improved radius allows us to recover the degree 1 polynomial $P_1(X)$. At the i^{th} stage, we have already recovered $P_0(X)$, $P_1(X)$, $\ldots$, $P_{i-1}(X)$. Consider, for each $\alpha \in S$, the function $f_\alpha(Y) = r(\alpha, Y) - \sum_{j=0}^{i-1} P_j(\alpha)Y^{d-j}$. Our algorithm decodes $f_\alpha(Y)$ to the nearest degree $d - i$ polynomial: note that as i increases, we are decoding to a lower degree polynomial, and hence we are able to handle a larger fraction of errors. Define $h(\alpha)$ to be the coefficient of Y^{d-i} in the polynomial so obtained; this "should" equal the evaluation of the degree i polynomial $P_i(\alpha)$. So we next decode $h(\alpha)$ to the nearest degree i polynomial (using the appropriate soft information), and it turns out that this decoded polynomial must equal $P_i(X)$. By the time i reaches d, we would have recovered $P_0(X), P_1(X), \ldots, P_d(X)$, and hence all of $P(X, Y)$. Summarizing, the algorithm repeatedly decodes the columns $r(\alpha, \cdot)$, and at each stage it uses the relationship between the different univariate polynomial $P(\alpha, Y)$ to: (1) learn a little bit more about the polynomial $P(X, Y)$, and (2) increase the radius to which we can decode $r(\alpha, \cdot)$ in the next stage. This completes the description of the algorithm in the $m = 2$ case.

The case of general m is very similar, with only a small augmentation needed. Decoding m-variate polynomials turns out to reduce to decoding $m - 1$-variate polynomials with soft information; thus in order to make a sustainable recursive algorithm, we aim a little higher and instead solve the more general problem of decoding multivariate polynomial codes with uncertainties (where each coordinate of the received word has an associated "confidence" level).

To implement the above algorithms in near-linear time, we use some tools from list-decoding. The main bottleneck in the running time is the requirement of having to decode the same column $r(\alpha, \cdot)$ multiple times to larger and larger radii (to lower and lower degree polynomials). To save on these decodings, we can instead list-decode $r(\alpha, \cdot)$ to a large radius using a near-linear time list-decoder for Reed-Solomon codes; this reduces the number of required decodings of the same column from d to $O(1)$ (provided $d < (1 - \Omega(1))|S|$). For the $m = 2$ case this works fine, but for $m > 2$ case this faces a serious obstacle; in general it is impossible to efficiently list-decode Reed-Solomon codes *with uncertainties* beyond half the minimum distance of the code (the list size can be superpolynomial). We get around this using some technical ideas, based on speeding-up the decoding of Reed-Muller codes with uncertainties when the fraction of errors is significantly smaller than half the minimum distance. For details, see Section 6.

1.4 Organization of this paper

In Section 2, we cover the notion of weighted distance, which will be used in handling Reed-Solomon and Reed-Muller decoding with soft information on the reliability of the symbols in the encoding. In Section 3, we state and prove a polynomial time algorithm for decoding bivariate Reed-Muller codes to half the minimum distance. We then generalize the proof to decode multivariate Reed-Muller codes in Section 4. Finally, in sections 5 and 6, we show that decoding Reed-Muller codes to almost half the minimum distance can be done in near-linear time by improving on the algorithms in Section 3 and 4.

2 Preliminaries

At various stages of the decoding algorithm, we will need to deal with symbols and received words in which we have varying amounts of confidence. We now introduce some language to deal with such notions.

Let Σ denote an alphabet. A *weighted symbol* of Σ is simply an element of $\Sigma \times [0,1]$. In the weighted symbol (σ, u), we will be thinking of $u \in [0,1]$ as our uncertainty that σ is the symbol we should be talking about.

For a weighted symbol (σ, u) and a symbol σ', we define their distance $\Delta((\sigma,u),\sigma')$ by:

$$\Delta((\sigma,u),\sigma') = \begin{cases} 1-u/2 & \sigma \neq \sigma' \\ u/2 & \sigma = \sigma' \end{cases}$$

For a weighted function $r : T \to \Sigma \times [0,1]$, and a (conventional) function $f : T \to \Sigma$, we define their Hamming distance by

$$\Delta(r,f) = \sum_{t \in T} \Delta(r(t), f(t)).$$

The key inequality here is the triangle inequality.

▶ **Lemma 2.1** (Triangle inequality for weighted functions). *Let $f, g : T \to \Sigma$ be functions, and let $r : T \to \Sigma \times [0,1]$ be a weighted function. Then:*

$$\Delta(r,f) + \Delta(r,g) \geq \Delta(f,g).$$

Proof. We will show that if $t \in T$ is such that $f(t) \neq g(t)$, then $\Delta(r(t), f(t)) + \Delta(r(t), g(t)) \geq 1$. This will clearly suffice to prove the lemma.

Let $r(t) = (\sigma, u)$. Suppose $f(t) = \sigma_1$ and $g(t) = \sigma_2$. Then either $\sigma \neq \sigma_1$ or $\sigma \neq \sigma_2$, or both. Thus either we have $\Delta(r(t), f(t)) + \Delta(r(t), g(t)) = (1 - u/2) + u/2$ or we have $\Delta(r(t), f(t)) + \Delta(r(t), g(t)) = u/2 + (1 - u/2)$, or we have $\Delta(r(t), f(t)) + \Delta(r(t), g(t)) = (1 - u/2) + (1 - u/2)$. In all cases, we have $\Delta(r(t), f(t)) + \Delta(r(t), g(t)) \geq 1$, as desired. ◀

The crucial property that this implies is the unique decodability up to half the minimum distance of a code for *weighted* received words.

▶ **Lemma 2.2.** *Let $\mathcal{C} \subseteq \Sigma^T$ be a code with minimum distance Δ. Let $r : T \to \Sigma \times [0,1]$ be a weighted function. Then there is at most one $f \in \mathcal{C}$ satisfying*

$$\Delta(r,f) < \Delta/2.$$

Furthermore, for this particular definition of weighted distance, there is a very natural decoding algorithm, due to Forney, to find the unique $f \in \mathcal{C}$ in Lemma 2.2 [13]. For each weighted symbol (x, u), we erase x with probability u. We then apply a standard decoding algorithm that handles both errors and erasures. This successfully finds the unique codeword f as long as $2E + F < \Delta$, where E denotes the number of errors and F denotes the number of erasures. With this definition of weighted distance, the condition that $\Delta(r,f) < \Delta/2$ is equivalent to the expected value of $2E + F$ being at most Δ.

3 Bivariate Reed-Muller Decoding

In this section, we provide an algorithm for decoding bivariate Reed-Muller codes to half the minimum distance. Consider the bivariate Reed-Muller decoding problem. We are given a received word $r : S^2 \to \mathbb{F}$. Suppose that there is a codeword $C \in \mathbb{F}[X, Y]$ with $\deg(C) \leq d$, whose distance $\Delta(r, C)$ from the received word is at most half the minimum distance $|S|(|S| - d)/2$. The following result says that there is a polynomial time algorithm in the size of the input $|S|^2$ to find C:

▶ **Theorem 3.1.** *Let $\mathbb{F}$ be a finite field and let $S \subseteq \mathbb{F}$ be a nonempty subset of size $|S| = n$. Given a received word $r : S^2 \to \mathbb{F}$, there is a $O(n^3 \operatorname{polylog}(n, |\mathbb{F}|))$ time algorithm to find the unique polynomial (if it exists) $C \in \mathbb{F}[X, Y]$ with $\deg(C) \leq d$ such that*

$$\Delta(r, C) < \frac{n^2}{2}\left(1 - \frac{d}{n}\right).$$

3.1 Outline of Algorithm

The general idea of the algorithm is to write $C(X, Y) = \sum_{i=0}^{d} P_i(X)Y^{d-i} \in \mathbb{F}[X][Y]$ as a polynomial in Y with coefficients as polynomials in $\mathbb{F}[X]$, and attempt to uncover the coefficients $P_i(X)$ one at a time.

We outline the first iteration of the algorithm, which uncovers the coefficient $P_0(X)$ of degree 0. View the encoded message as a matrix on $S \times S$, where the rows are indexed by $x \in S$ and the columns by $y \in S$. We first Reed-Solomon decode the rows $r(x, Y), x \in S$ to half the minimum distance $(n - d)/2$ and extract the coefficient of Y^d in those decodings. This gives us guesses for what $P_0(x)$ is for $x \in S$. However, this isn't quite enough to determine $P_0(X)$. So we will also include some soft information which tells us how uncertain we are that the coefficient is correct. The uncertainty is a number in $[0, 1]$ that is based on how far the decoded codeword $G_x(Y)$ is from the received word $r(x, Y)$. The farther apart, the higher the uncertainty. A natural choice for the uncertainty is simply the ratio of the distance $\Delta(G_x(Y), r(x, Y))$ to half the minimum distance $(n - d)/2$. In the event that the Reed-Solomon decoding finds no codeword, we make an arbitrary guess and set the uncertainty to be 1. Let $f : S \to F \times [0, 1]$ be the function of guesses for $P_0(x)$ and their uncertainties. We then use a Reed-Solomon decoder with uncertainties to find the degree 0 polynomial that is closest to $f(X)$. This will give us $P_0(X)$. Finally, subtract $P_0(X)Y^d$ from $r(X, Y)$ and repeat to get the subsequent coefficients.

In the algorithm, we use REED-SOLOMON-DECODER(r, d) to denote the $O(n \operatorname{polylog} n)$ time algorithm that performs Reed-Solomon decoding of degree d to half the minimum distance [11, 12]. We use RS-SOFT-DECODER(r, d) to denote the $O(n^2 \operatorname{polylog} n)$ time algorithm that performs Reed-Solomon decoding of degree d with uncertainties to half the minimum distance, which is based on Forney's generalized minimum distance decoding algorithm for concatenated codes [13].

3.2 Proof of Theorem 3.1

Correctness of Algorithm. It suffices to show that $Q_i(X) = P_i(X)$ for $i = 0, 1, \ldots, d$, which we prove by induction. For this proof, the base case and inductive step can be handled by a single proof. We assume the inductive hypothesis that we have $Q_j(X) = P_j(X)$ for $j < i$. Note that the base case is $i = 0$ and in this case, we assume nothing.

It is enough to show $\Delta(f_i(X), P_i(X)) < \frac{n}{2}\left(1 - \frac{i}{n}\right)$. Then $P_i(x)$ is the unique polynomial within weighted distance $\frac{n}{2}\left(1 - \frac{i}{n}\right)$ of $f_i(X)$. So RS-SOFT-DECODER$(f_i(X), i)$ will output $Q_i(X) = P_i(X)$.

Algorithm 1 Decoding Bivariate Reed-Muller

Input: $r : S^2 \to \mathbb{F}$.
for $i = 0, 1, \ldots, d$ **do**
  Define $r_i : S \times S \to \mathbb{F}$ by

$$r_i(X, Y) = r(X, Y) - \sum_{j=0}^{i-1} Q_j(X) Y^{d-j}.$$

  for $x \in S$ **do**
    Define $r_{i,x} : S \to \mathbb{F}$ by

$$r_{i,x}(Y) = r_i(x, Y).$$

    Define $G_x(Y) \in \mathbb{F}[Y]$ by

$$G_x(Y) = \text{REED-SOLOMON-DECODER}(r_{i,x}(Y), d - i).$$

    $\sigma_x \leftarrow \text{Coeff}_{Y^{d-i}}(G_x)$.
    $\delta_x \leftarrow \Delta(r_{i,x}, G_x)$.
  end for
  Define the weighted function $f_i : S \to \mathbb{F} \times [0, 1]$ by

$$f_i(x) = \left(\sigma_x, \frac{\delta_x}{(n - d + i)/2}\right).$$

  Define $Q_i : S \to \mathbb{F}$ by

$$Q_i(X) = \text{RS-SOFT-DECODER}(f_i(X), i).$$

end for
Output: $\sum_{i=0}^{d} Q_i(X) Y^{d-i}$.

We first show that $r_i(X, Y)$ is close to $C_i(X, Y) = \sum_{j=i}^{d} P_j(X) Y^{d-j}$. Observe that:

$$\begin{aligned}
&r_i(X, Y) - C_i(X, Y) \\
&= (r_i(X, Y) + \sum_{j=1}^{i-1} P_j(X) Y^{d-j}) - (C_i(X, Y) + \sum_{j=1}^{i-1} P_j(X) Y^{d-j})) \\
&= (r_i(X, Y) + \sum_{j=1}^{i-1} Q_j(X) Y^{d-j}) - C(X, Y) \\
&= r(X, Y) - C(X, Y).
\end{aligned}$$

Hence,

$$\Delta(r_i(X, Y), C_i(X, Y)) = \Delta(r(X, Y), C(X, Y)) < \frac{n^2}{2}\left(1 - \frac{d}{n}\right).$$

For each $x \in S$, define $C_{i,x}(Y) = C_i(x, Y)$. Define $\Delta_x = \Delta(r_{i,x}(Y), C_{i,x}(Y))$. Let $A = \{x \in S | G_x(Y) = C_{i,x}(Y)\}$ be the set of choices of x such that $G_x(Y) = \text{REED-SOLOMON-DECODER}(r_{i,x}(Y), d - i)$ produces $C_{i,x}(Y)$.

Then, for $x \in A$, we have

$$\delta_x = \Delta(r_{i,x}(Y), G_x(Y)) = \Delta(r_{i,x}(Y), C_{i,x}(Y)) = \Delta_x,$$

which gives us an uncertainty value of

$$u_{i,x} = \frac{\Delta_x}{(n-d+i)/2}.$$

For $x \notin A$, either we have $G_x \neq C_{i,x}$, or the Reed-Solomon decoder does not find a polynomial. In the first case, Lemma 2.1 tells us:

$$\delta_x = \Delta(r_{i,x}(Y), G_x(Y)) \geq n - d + i - \Delta(r_{i,x}(Y), C_{i,x}(Y)) = n - d + i - \Delta_x,$$

which gives us an uncertainty value of

$$u_{i,x} = \frac{n-d+i-\Delta_x}{(n-d+i)/2}.$$

Finally, in the case where the Reed-Solomon decoder does not find a polynomial, we get an uncertainty value of

$$u_{i,x} = 1.$$

This means that the contribution of the corresponding guess to the weighted distance $\Delta(f_i(X), P_i(X))$ is $1/2$, However, we know that since no polynomial was found, $\Delta_x \geq \frac{n-d+i}{2}$, so the contribution to the weighted distance had the Reed-Solomon decoder found an incorrect polynomial not matching the true codeword is $1 - \frac{1}{2}\frac{n-d+i-\Delta_x}{(n-d+i)/2} \geq 1/2$. So for the purposes of upper bounding the weighted distance $\Delta(f_i(X), P_i(X))$, we treat this case the same as decoding to the wrong polynomial.

We now upper bound $\Delta(f_i(X), P_i(X))$:

$$\begin{aligned}
\Delta(f_i(X), P_i(X)) &\leq \sum_{x \in A} \frac{1}{2}\frac{\Delta_x}{(n-d+i)/2} + \sum_{x \notin A} 1 - \frac{1}{2}\frac{n-d+i-\Delta_x}{(n-d+i)/2} \\
&\leq \sum_{x \in A} \frac{\Delta_x}{n-d+i} + \sum_{x \notin A} 1 - \frac{n-d+i-\Delta_x}{n-d+i} \\
&= \sum_{x \in A} \frac{\Delta_x}{n-d+i} + \sum_{x \notin A} \frac{\Delta_x}{n-d+i} \\
&= \sum_{x \in S^m} \frac{\Delta_x}{n-d+i} \\
&= \frac{\Delta(r_i(X,Y), C_i(X,Y))}{n-d+i} \\
&< \frac{n^2}{2}\left(1 - \frac{d}{n}\right)\frac{1}{n-d+i} \\
&= \frac{n}{2} \cdot \frac{n-d}{n-d+i} \\
&\leq \frac{n}{2} \cdot \frac{n-i}{n} \\
&= \frac{n}{2}\left(1 - \frac{i}{n}\right).
\end{aligned}$$

Runtime of Algorithm

We claim that the runtime of our algorithm is $O(n^3 \operatorname{polylog} n)$, ignoring the $\operatorname{polylog} |\mathbb{F}|$ factor from field operations. The algorithm has $d+1$ iterations. In each iteration, we update r_i, apply REED-SOLOMON-DECODER to n rows and apply RS-SOFT-DECODER a single time to get the leading coefficient. As updating takes $O(n^2)$ time, REED-SOLOMON-DECODER takes $O(n \operatorname{polylog} n)$ time, and RS-SOFT-DECODER takes $O(n^2 \operatorname{polylog} n)$ time, we get $O(n^2 \operatorname{polylog} n)$ for each iteration. $d+1$ iterations gives a total runtime of $O(dn^2 \operatorname{polylog} n) < O(n^3 \operatorname{polylog} n)$. ◀

4 Reed-Muller Decoding for General m

We now generalize the algorithm for decoding bivariate Reed-Muller codes to handle Reed-Muller codes of any number of variables. As before, we write the codeword as a polynomial in one of the variables and attempt to uncover its coefficients one at a time. Interestingly, this leads us to a Reed-Muller decoding on one fewer variable, but with uncertainties. This lends itself nicely to an inductive approach on the number of variables, however, the generalization requires us to be able to decode Reed-Muller codes with uncertainties. This leads us to our main theorem:

▶ **Theorem 4.1.** *Let $\mathbb{F}$ be a finite field and let $S \subseteq \mathbb{F}$ be a nonempty subset of size $|S| = n$. Given a received word with uncertainties $r : S^m \to \mathbb{F} \times [0,1]$, there is a $O(n^{m+2} \operatorname{polylog}(n, |\mathbb{F}|))$ time algorithm to find the unique polynomial (if it exists) $C \in \mathbb{F}[X_1, \ldots, X_m]$ with $\deg(C) \leq d$ such that*

$$\Delta(r, C) < \frac{n^m}{2}\left(1 - \frac{d}{n}\right).$$

Note that to decode a Reed-Muller code without uncertainties, we may just set all the initial uncertainties to 0. The algorithm slows by a factor of n from the bivariate case due to having to use the RS-SOFT-DECODER instead of the faster REED-SOLOMON-DECODER on the rows of the received word.

Proof. The proof is by induction on the number of variables, and closely mirrors the proof of the bivariate case.

Base Case

We are given a received word with uncertainties $r : S \to \mathbb{F} \times [0,1]$ and asked to find the unique polynomial $C \in \mathbb{F}[X]$ with $\deg(C) \leq d$ within weighted distance $\frac{n-d}{2}$ of r. This is just Reed-Solomon decoding with uncertainty, which can be done in time $O(n^2 \operatorname{polylog} n)$.

Inductive Step

Assume that the result holds for m variables. That is, assume we have access to an algorithm REED-MULLER-DECODER(r, m, d) which takes as input a received word with uncertainties $r : S^m \to \mathbb{F} \times [0,1]$, and outputs the unique polynomial of degree at most d (if it exists) within weighted distance $\frac{n^m}{2}\left(1 - \frac{d}{n}\right)$ from r. We want to produce an algorithm for $m+1$ variables. Before we progress, we set up some definitions to make the presentation and analysis of the algorithm cleaner. We are given $r : S^{m+1} \to \mathbb{F} \times [0,1]$. View r as a map from $S^m \times S \to \mathbb{F} \times [0,1]$, and write $r(\boldsymbol{X}, Y) = (\overline{r}(\boldsymbol{X}, Y), u(\boldsymbol{X}, Y))$.

Suppose that there exists a polynomial $C \in \mathbb{F}[\boldsymbol{X}, Y]$ with $\deg(C) \leq d$ such that

$$\Delta(r, C) < \frac{n^{m+1}}{2}\left(1 - \frac{d}{n}\right).$$

View C as a polynomial in Y with coefficients in $\mathbb{F}[\boldsymbol{X}]$, $C(\boldsymbol{X}, Y) = \sum_{i=0}^{d} P_i(\boldsymbol{X})Y^{d-i}$. The general strategy of the algorithm is to determine the P_i's inductively by performing $d+1$ iterations from $i = 0$ to $i = d$, and recovering $P_i(\boldsymbol{X})$ at the i-th iteration.

For the i-th iteration, consider the word

$$r_i(\boldsymbol{X}, Y) = \left(\overline{r}(\boldsymbol{X}, Y) - \sum_{j=0}^{i-1} P_j(\boldsymbol{X})Y^{d-j}, u(\boldsymbol{X}, Y)\right).$$

Since r is close to $\sum_{j=0}^{d} P_j(\boldsymbol{X})Y^{d-j}$, r_i will be close to $C_i = \sum_{j=i}^{d} P_j(\boldsymbol{X})Y^{d-j}$. Our goal is to find $P_i(\boldsymbol{X})$, the leading coefficient of C_i when viewed as a polynomial in Y. For each $\boldsymbol{x} \in S^m$, we decode the Reed-Solomon code with uncertainties given by $r_i(\boldsymbol{x}, Y)$ and extract the coefficient of Y^{d-i} along with how uncertain we are about the correctness of this coefficient. This gives us a guess for the value $P_i(\boldsymbol{x})$ and our uncertainty for this guess. We construct the function $f_i : S^m \to F \times [0, 1]$ of guesses for P_i with their uncertainties. We then apply the induction hypothesis of Theorem 4.1 to f_i to recover P_i.

Correctness of Algorithm

Suppose there is a polynomial $C(\boldsymbol{X}, Y) = \sum_{i=0}^{d} P_i(\boldsymbol{X})Y^{d-i}$ such that

$$\Delta(r(\boldsymbol{X}, Y), C(\boldsymbol{X}, Y)) < \frac{n^{m+1}}{2}\left(1 - \frac{d}{n}\right).$$

We will show by induction that the i-th iteration of the algorithm produces $Q_i(\boldsymbol{X}) = P_i(\boldsymbol{X})$. For this proof, the base case and inductive step can be handled by a single proof. We assume the inductive hypothesis that we have $Q_j(\boldsymbol{X}) = P_j(\boldsymbol{X})$ for $j < i$. Note that the base case is $i = 0$ and in this case, we assume nothing.

It is enough to show $\Delta(f_i(\boldsymbol{X}), P_i(\boldsymbol{X})) < \frac{n^m}{2}\left(1 - \frac{i}{n}\right)$. Then $P_i(\boldsymbol{X})$ is the unique polynomial within weighted distance $\frac{n^m}{2}\left(1 - \frac{i}{n}\right)$ of $f_i(\boldsymbol{X})$. This means that REED-MULLER-DECODER$(f_i(\boldsymbol{X}), m, i)$ will output $Q_i(\boldsymbol{X}) = P_i(\boldsymbol{X})$.

We first show that $r_i(\boldsymbol{X}, Y)$ is close to $C_i(\boldsymbol{X}, Y) = \sum_{j=i}^{d} P_j(\boldsymbol{X})Y^{d-j}$. Observe that:

$$\begin{aligned}
&r_i(\boldsymbol{X}, Y) - C_i(\boldsymbol{X}, Y) \\
&= (r_i(\boldsymbol{X}, Y) + \sum_{j=1}^{i-1} P_j(\boldsymbol{X})Y^{d-j}) - (C_i(\boldsymbol{X}, Y) + \sum_{j=1}^{i-1} P_j(\boldsymbol{X})Y^{d-j})) \\
&= (r_i(\boldsymbol{X}, Y) + \sum_{j=1}^{i-1} Q_j(\boldsymbol{X})Y^{d-j}) - C(\boldsymbol{X}, Y) \\
&= r(\boldsymbol{X}, Y) - C(\boldsymbol{X}, Y).
\end{aligned}$$

Hence,

$$\Delta(r_i(\boldsymbol{X}, Y), C_i(\boldsymbol{X}, Y)) = \Delta(r(\boldsymbol{X}, Y), C(\boldsymbol{X}, Y)) < \frac{n^{m+1}}{2}\left(1 - \frac{d}{n}\right).$$

Algorithm 2 Decoding Reed-Muller with Uncertainties

Input: $r : S^{m+1} \to \mathbb{F} \times [0,1]$.
for $i = 0, 1, \ldots, d$ **do**
  Define $r_i : S^m \times S \to \mathbb{F} \times [0,1]$ by

$$r_i(\boldsymbol{X}, Y) = \left(\overline{r}(\boldsymbol{X}, Y) - \sum_{j=0}^{i-1} Q_j(\boldsymbol{X}) Y^{d-j}, u(\boldsymbol{X}, Y) \right).$$

  for $\boldsymbol{x} \in S^m$ **do**
    Define $r_{i,\boldsymbol{x}} : S \to \mathbb{F} \times [0,1]$ by

$$r_{i,\boldsymbol{x}}(Y) = r_i(\boldsymbol{x}, Y).$$

    Define $G_{\boldsymbol{x}}(Y) \in \mathbb{F}[Y]$ by

$$G_{\boldsymbol{x}}(Y) = \text{RS-SOFT-DECODER}(r_{i,\boldsymbol{x}}(Y), d-i).$$

    $\sigma_{\boldsymbol{x}} \leftarrow \text{Coeff}_{Y^{d-i}}(G_{\boldsymbol{x}})$.
    $\delta_{\boldsymbol{x}} \leftarrow \Delta(r_{i,\boldsymbol{x}}, G_{\boldsymbol{x}})$.
  end for
  Define the weighted function $f_i : S^m \to \mathbb{F} \times [0,1]$ by

$$f_i(\boldsymbol{x}) = \left(\sigma_{\boldsymbol{x}}, \frac{\delta_{\boldsymbol{x}}}{(n-d+i)/2} \right).$$

  Define $Q_i : S^m \to \mathbb{F}$ by

$$Q_i(\boldsymbol{X}) = \text{REED-MULLER-DECODER}(f_i(\boldsymbol{X}), m, i).$$

end for
Output: $\sum_{i=0}^{d} Q_i(\boldsymbol{X}) Y^{d-i}$.

For each $\boldsymbol{x} \in S^m$, define $C_{i,\boldsymbol{x}}(Y) = C_i(\boldsymbol{x}, Y)$. Define $\Delta_{\boldsymbol{x}} = \Delta(r_{i,\boldsymbol{x}}(Y), C_{i,\boldsymbol{x}}(Y))$. Let $A = \{\boldsymbol{x} \in S^m | G_{\boldsymbol{x}}(Y) = C_{i,\boldsymbol{x}}(Y)\}$ be the set of choices of $\boldsymbol{x}$ such that $G_{\boldsymbol{x}}(Y) = \text{RS-SOFT-DECODER}(r_{i,\boldsymbol{x}}(Y), d-i)$ produces $C_{i,\boldsymbol{x}}(Y)$.

Then, for $\boldsymbol{x} \in A$, we have

$$\delta_{\boldsymbol{x}} = \Delta(r_{i,\boldsymbol{x}}(Y), G_{\boldsymbol{x}}(Y)) = \Delta(r_{i,\boldsymbol{x}}(Y), C_{i,\boldsymbol{x}}(Y)) = \Delta_{\boldsymbol{x}}.$$

And for $\boldsymbol{x} \notin A$, we have $G_{\boldsymbol{x}} \neq C_{i,\boldsymbol{x}}$, so

$$\delta_{\boldsymbol{x}} = \Delta(r_{i,\boldsymbol{x}}(Y), G_{\boldsymbol{x}}(Y)) \geq n - d + i - \Delta(r_{i,\boldsymbol{x}}(Y), C_{i,\boldsymbol{x}}(Y)) = n - d + i - \Delta_{\boldsymbol{x}}.$$

We now upper bound $\Delta(f_i(\boldsymbol{X}), P_i(\boldsymbol{X}))$:

$$\begin{aligned}
\Delta(f_i(\boldsymbol{X}), P_i(\boldsymbol{X})) &\leq \sum_{\boldsymbol{x} \in A} \frac{1}{2} \frac{\delta_{\boldsymbol{x}}}{(n-d+i)/2} + \sum_{\boldsymbol{x} \notin A} 1 - \frac{1}{2} \frac{\delta_{\boldsymbol{x}}}{(n-d+i)/2} \\
&\leq \sum_{\boldsymbol{x} \in A} \frac{\Delta_{\boldsymbol{x}}}{n-d+i} + \sum_{\boldsymbol{x} \notin A} 1 - \frac{n-d+i-\Delta_{\boldsymbol{x}}}{n-d+i}
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{\boldsymbol{x} \in A} \frac{\Delta_{\boldsymbol{x}}}{n-d+i} + \sum_{\boldsymbol{x} \notin A} \frac{\Delta_{\boldsymbol{x}}}{n-d+i} \\
&= \sum_{\boldsymbol{x} \in S^m} \frac{\Delta_{\boldsymbol{x}}}{n-d+i} \\
&= \frac{\Delta(r_i(\boldsymbol{X}, Y), C_i(\boldsymbol{X}, Y))}{n-d+i} \\
&< \frac{n^{m+1}}{2} \left(1 - \frac{d}{n}\right) \frac{1}{n-d+i} \\
&= \frac{n^m}{2} \cdot \frac{n-d}{n-d+i} \\
&\leq \frac{n^m}{2} \cdot \frac{n-i}{n} \\
&= \frac{n^m}{2} \left(1 - \frac{i}{n}\right).
\end{aligned}
$$

Runtime of Algorithm

We claim the runtime of our m-variate Reed-Muller decoder is $O(n^{m+2} \operatorname{polylog} n)$, ignoring the $\operatorname{polylog} |\mathbb{F}|$ factor from field operations. We again proceed by induction on m. In the base case of $m = 1$, we simply run the Reed-Solomon decoder with uncertainties, which runs in $O(n^2 \operatorname{polylog} n)$ time. Now suppose the m-variate Reed-Muller decoder runs in time $O(n^{m+2} \operatorname{polylog} n)$. We need to show that the $m+1$-variate Reed-Muller decoder runs in time $O(n^{m+3} \operatorname{polylog} n)$.

The algorithm makes $d+1$ iterations. In each iteration, we perform n^m Reed-Solomon decodings with uncertainties, and extract the leading coefficient along with its uncertainty for each one. Each Reed-Solomon decoding takes $O(n^2 \operatorname{polylog} n)$ time, while computing an uncertainty of a leading coefficient takes $O(n \operatorname{polylog} n)$. So in this step, we have cumulative runtime $O(n^{m+2} \operatorname{polylog} n)$. Next we do a single m-variate Reed-Muller decoding with uncertainties, which takes $O(n^{m+2} \operatorname{polylog} n)$ by our induction hypothesis. This makes the total runtime $O(dn^{m+2} \operatorname{polylog} n) \leq O(n^{m+3} \operatorname{polylog} n)$, as desired. ◀

5 Near-Linear Time Decoding in the Bivariate Case

In this section, we present our near-linear time decoding algorithm for bivariate Reed-Muller codes.

▶ **Theorem 5.1.** *Let $\alpha \in (0,1)$ be a constant. Let $\mathbb{F}$ be a finite field and let $S \subseteq \mathbb{F}$ be a nonempty subset of size $|S| = n$. Let $d = \alpha n$. Given a received word $r : S^2 \to \mathbb{F}$, there is a $O(n^2 \operatorname{polylog}(n, |\mathbb{F}|))$ time algorithm to find the unique polynomial (if it exists) $C \in \mathbb{F}[X, Y]$ with $\deg(C) \leq d$ such that*

$$\Delta(r, C) < \frac{n^2}{2} \left(1 - \frac{d}{n} - \frac{1}{\sqrt{n}}\right).$$

5.1 Outline of Improved Algorithm

Recall that the decoding algorithms we presented in the previous sections make $d+1$ iterations, where $d = \alpha n$, revealing a single coefficient of the nearest codeword during one iteration. In a given iteration, we decode each row of $r_i(X, Y)$ to the nearest polynomial of degree $d - i$,

extracting the coefficient of Y^{d-i} and its uncertainty. Then we Reed-Solomon decode with uncertainties to get the leading coefficient of $C(X, Y)$, when viewed as a polynomial in Y.

The runtime of this algorithm is $O(n^3 \operatorname{polylog} n)$. Each iteration has n Reed-Solomon decodings and a single Reed-Solomon decoding with uncertainties. As Reed-Solomon decoding takes $O(n \operatorname{polylog} n)$ time and Reed-Solomon decoding with uncertainties takes $O(n^2 \operatorname{polylog} n)$ time, we get a runtime of $O(n^3 \operatorname{polylog} n)$ with $d+1$ iterations. To achieve near-linear time, we need to shave off a factor of n on both the number of Reed-Solomon decodings and the runtime of Reed-Solomon decoding with uncertainties.

To save on the number of Reed-Solomon decodings, we will instead list decode beyond half the minimum distance (using a near-linear time Reed-Solomon list-decoder), and show that the list we get is both small and essentially contains all of the decoded polynomials we require for $\Omega(n)$ iterations of i. So we will do $O(n)$ Reed-Solomon list-decodings total instead of $O(n^2)$ Reed-Solomon unique decodings to half the minimum distance.

To save on the runtime of Reed-Solomon decoding with uncertainties, we will use a probabilistic variant of Forney's generalized minimum distance decoding algorithm, which runs in near-linear time, but reduces the decoding radius from $1/2$ the minimum distance to $1/2 - o(1)$ of the minimum distance.

5.2 Proof of Fast Bivariate Reed-Muller Decoding

Proof of Theorem 5.1. As in the proof of Theorem 3.1, we write $C = \sum_{j=0}^{d} P_j(X) Y^{d-j}$, and inductively find the $P_i(X)$. Suppose that we have successfully found the first i of the $P_j(X)$ and are now trying to find $P_i(X)$. Also as before, we fix $x \in S$ and guess the value of $P_i(x)$ by Reed-Solomon decoding $r_{i,x} = r(x, Y) - \sum_{j=0}^{i-1} P_j(x) Y^{d-j}$ to the nearest polynomial of degree at most i within distance $(n - d + i)/2$.

Reducing the Number of Decodings

To reduce the number of Reed-Solomon decodings, we will instead list decode past half the minimum distance, and use the small list of polynomials we get to guess $P_i(x)$ for the next $\Omega(n)$ values of j. In the above setting, we have that $r_{i,x} : S \to \mathbb{F}$ is a received word for a Reed-Solomon code $\mathcal{C}_i$ of degree at most $d_i = d - i$. Let t be the radius to which we list decode, and let $L_{i,x} = \{C \in \mathcal{C}_i | \Delta(C, r_{i,x}) < t\}$ be the list of codewords within distance t of $r_{i,x}$. The radius to which we can decode while maintaining a polynomial-size list is given by the Johnson bound:

$$n(1 - \sqrt{1 - \delta_i}),$$

where $\delta_i = 1 - \frac{d-i}{n} > 1 - \frac{d}{n} = 1 - \alpha$ is the relative distance of the code. By Taylor approximating the square root, we see that the Johnson bound exceeds half the minimum distance by $\Omega(n)$:

$$\begin{aligned} n(1 - \sqrt{1 - \delta_i}) &> n(1 - (1 - \delta_i/2 + \delta_i^2/8 + 3\delta_i^3/16)) \\ &= n(\delta_i/2 + (1-\alpha)^2/8 + 3(1-\alpha)^3/16) \\ &= (n - d + i)/2 + ((1-\alpha)^2/8)n + cn, \end{aligned}$$

where $c = 3(1-\alpha)^3/16$ is a positive constant. By a standard list-size bound as in the one in Cassuto and Bruck [14], we see that if we set the list decoding radius $t = (n - d + i)/2 + ((1-\alpha)^2/8)n$, then the size of the list $|L_{i,x}| < \frac{1}{c}$ is constant. So the list decoding radius exceeds half the minimum distance by $\Omega(n)$, and the list size is constant. By Aleknovich's

fast algorithm for weighted polynomial construction [15], the list $L_{i,x}$ can be produced in time $(1/\alpha)^{O(1)} n \log^2 n \log\log n = O(n \operatorname{polylog} n)$. We will let RS-LIST-DECODER(r, d, t) denote the Reed-Solomon list decoder that outputs a list of all ordered pairs of polynomials of degree at most d within distance t to the received word r along with their distances to r. Since the list size is constant, all of the distances can be computed in $O(n \operatorname{polylog} n)$ time.

For the next cn values of j, we search the $O(1)$-size list $L_{i,x}$ to find the nearest polynomial of degree at most $n - d + j$ within distance $(n - d + j)/2$ from $r_{j,x}$.

Faster Reed-Solomon Decoding with Uncertainties

Once we have all the guesses for $P_i(x), x \in S$ along with their uncertainties, we want to apply a near-linear time decoding algorithm to find $P_i(x)$. In the appendix, we give a description of the probabilistic GMD algorithm that gives a faster Reed-Solomon decoder with uncertainties. We will refer to this algorithm as FAST-RS-DECODER(f, i), where $f : S \to \mathbb{F} \times [0, 1]$ is a received word with uncertainties, and i is the degree of the code. FAST-RS-DECODER(f, i) will output the codeword within distance $(n - i - \sqrt{n})/2$ (if it exists) with probability at least $1 - \frac{1}{n^{\Omega(1)}}$ (the $\Omega(1)$ can be chosen to be an arbitrary constant, by simply repeating the algorithm independently several times).

Correctness of Algorithm

View the received word as a matrix on $S \times S$, where the rows are indexed by $x \in S$ and the columns by $y \in S$. For correctness, we have to show two things. First, that Algorithm 3 produces the same row decodings $G_x(Y)$ as Algorithm 2. Second, that the algorithm actually extracts the coefficients of $C(X, Y) = \sum_{i=0}^{d} P_i(X) Y^{d-i}$ when viewed as a polynomial in Y, i.e. $Q_i(X) = P_i(X)$ for $i = 0, \ldots, d$. Define $r_{j \cdot 2cn+k} : S \times S \to \mathbb{F}$ by

$$r_{j \cdot 2cn+k}(X, Y) = r(X, Y) - \sum_{i=0}^{j \cdot 2cn+k-1} Q_i(X) Y^{d-i},$$

and define $r_{j \cdot 2cn+k,x} : S \to \mathbb{F}$ by

$$r_{j \cdot 2cn+k,x}(Y) = r_{j \cdot 2cn+k}(x, Y).$$

Then we want to show that in each of the $d + 1$ iterations of (j, k), we have

$$G_x(Y) = \text{REED-SOLOMON-DECODER}\left(r_{j \cdot 2cn+k,x}(Y), d - j \cdot 2cn - k\right).$$

It is enough to instead show that the list $L_{j,k,x}$ contains all the polynomials of degree at most $d - j \cdot 2cn - k$ within distance $t_j = (n - d + j \cdot 2cn)/2 + cn > (n - d + j \cdot 2cn + k)/2$ of $r_{j \cdot 2cn+k,x}(Y)$. Furthermore, we want to show $Q_{j \cdot 2cn+k}(X) = P_{j \cdot 2cn+k}(X)$.

We prove this by induction on (j, k). The base case is $j = k = 0$. For each row $x \in S$, we have

$$L_{0,0,x} = \text{RS-LIST-DECODER}(r_{j \cdot 2cn,x}(Y), d - j \cdot 2cn, t_0).$$

The induction hypothesis is that for every $(j', k') < (j, k)$ in the lexicographic order, we have $L_{j',k',x} = \{(\overline{C}, \Delta(\overline{C}, r_{j' \cdot 2cn+k',x})) | \overline{C} \in \mathcal{C}_{j' \cdot 2cn+k'}, \Delta(\overline{C}, r_{j' \cdot 2cn+k',x}) < t_{j'}\}$ and that $Q_{j' \cdot 2cn+k'}(X) = P_{j' \cdot 2cn+k'}(X)$. We will show the corresponding statements hold true for (j, k).

If $k = 0$, then the fact that the algorithm extracted the correct coefficients thus far means that the $r_{j \cdot 2cn}$ are the same in both Algorithm 2 and Algorithm 3. Since $L_{j,0,x} =$

Algorithm 3 Decoding Bivariate Reed-Muller

Input: $r : S^2 \to \mathbb{F}$.
Let $c = ((1-\alpha)^2/8)$.
for $j = 0, 1, \ldots, \frac{d}{2cn}$ **do**
  Let $t_j = \frac{n-d+j\cdot 2cn}{2} + cn$.
  Define $r_{j\cdot 2cn} : S \times S \to \mathbb{F}$ by

$$r_{j\cdot 2cn}(X,Y) = r(X,Y) - \sum_{i=0}^{j\cdot 2cn-1} Q_i(X)Y^{d-i}.$$

  for $x \in S$ **do**
    Define $r_{j\cdot 2cn,x} : S \to \mathbb{F}$ by

$$r_{j\cdot 2cn,x}(Y) = r_{j\cdot 2cn}(x,Y).$$

    Define $\mathcal{C}_{j\cdot 2cn}$ by

$$\mathcal{C}_{j\cdot 2cn} = \{C(Y) \in \mathbb{F}[Y] | \deg(C) < d - j\cdot 2cn\}.$$

    Define $L_{j,0,x} = \text{RS-LIST-DECODER}(r_{j\cdot 2cn,x}(Y), d - j\cdot 2cn, t_j)$.
  end for
  for $k = 0, 1, \ldots, 2cn-1$ **do**
    for $x \in S$ **do**
      Define $(G_x(Y), \delta_x) \in L_{j,k,x}$ to be the unique codeword (if any) with

$$\delta_x < \frac{n-d+j\cdot 2cn+k}{2}$$

      $\sigma_x \leftarrow \text{Coeff}_{Y^{d-j\cdot 2cn-k}}(G_x)$.
    end for
    Define the weighted function $f_{j\cdot 2cn+k} : S \to \mathbb{F} \times [0,1]$ by

$$f_{j\cdot 2cn+k}(x) = \left(\sigma_x, \frac{\delta_x}{(n-d+j\cdot 2cn+k)/2}\right).$$

    Define $Q_{j\cdot 2cn+k} : S \to \mathbb{F}$ by

$$Q_{j\cdot 2cn+k}(X) = \text{FAST-RS-DECODER}(f_{j\cdot 2cn+k}(X), j\cdot 2cn+k).$$

    for $x \in S$ **do**
      Define

$$L_{j,k+1,x} = \{(C - Q_{j\cdot 2cn+k}(x)Y^{d-j\cdot 2cn-k}, \delta_{C,x}) \mid C \in L_{j,k,x}, \text{Coeff}_{Y^{d-j\cdot 2cn-k}}(C) = Q_{j\cdot 2cn+k}(x)\}.$$

    end for
  end for
end for
Output: $\sum_{i=0}^{d} Q_i(\boldsymbol{X})Y^{d-i}$.

RS-LIST-DECODER$(r_{j\cdot 2cn,x}(Y), d - j\cdot 2cn, t_j)$, the induction hypothesis on $L_{j,0,x}$ is met by the definition of RS-LIST-DECODER.

If $k \neq 0$, then we know from the induction hypothesis that

$$L_{j,k-1,x} = \{(\overline{C}, \Delta(\overline{C}, r_{j\cdot 2cn+k-1,x}))|\overline{C} \in \mathcal{C}_{j\cdot 2cn+k-1}, \Delta(\overline{C}, r_{j\cdot 2cn+k-1,x}) < t_j\}.$$

We want to say that

$$L_{j,k,x} = \{(\overline{C}, \Delta(\overline{C}, r_{j\cdot 2cn+k,x}))|\overline{C} \in \mathcal{C}_{j\cdot 2cn+k}, \Delta(\overline{C}, r_{j\cdot 2cn+k,x}) < t_j\}.$$

We defined $L_{j,k,x}$ in terms of $L_{j,k-1,x}$ to be:

$$\{(\overline{C} - Q_{j\cdot 2cn+k-1}(x)Y^{d-j\cdot 2cn-k+1}, \Delta(\overline{C}, r_{j\cdot 2cn+k-1,x})) \\ |\overline{C} \in L_{j,k-1,x}, \mathrm{Coeff}_{Y^{d-j\cdot 2cn-k+1}}(\overline{C}) = Q_{j\cdot 2cn+k-1}(x)\}.$$

As $Q_{j\cdot 2cn+k-1}(X) = P_{j\cdot 2cn+k-1}(X)$, $L_{j,k,x}$ is essentially obtained by taking the codewords with the correct leading coefficients and subtracting off the leading term. We claim that what we get is the set of all polynomials of degree at most $d - j\cdot 2cn - k$ within distance t_j of $r_{j\cdot 2cn+k,x}$.

Consider any $(G, \delta) \in L_{j,k,x}$. By definition of $L_{j,k,x}$, we know there exists a $\overline{C} \in L_{j,k-1,x}$ with $\mathrm{Coeff}_{Y^{d-j\cdot 2cn-k+1}}(\overline{C}) = Q_{j\cdot 2cn+k-1}(x)$ such that

$$(G, \delta) = (\overline{C} - Q_{j\cdot 2cn+k-1}(x)Y^{d-j\cdot 2cn-k+1}, \Delta(\overline{C}, r_{j\cdot 2cn+k-1,x})).$$

So we have

$$\begin{aligned}\overline{C} &= G + Q_{j\cdot 2cn+k-1}(x)Y^{d-j\cdot 2cn-k+1} \\ \delta &= \Delta(\overline{C}, r_{j\cdot 2cn+k-1,x}) < t_j.\end{aligned}$$

As $\mathrm{Coeff}_{Y^{d-j\cdot 2cn-k+1}}(\overline{C}) = Q_{j\cdot 2cn+k-1}(x)$, we have $\deg(G)$ is at most $d - j\cdot 2cn - k$. Also, as $r_{j\cdot 2cn+k-1,x} = r_{j\cdot 2cn+k,x} + Q_{j\cdot 2cn+k-1}(x)Y^{d-j\cdot 2cn-k+1}$, we have $\Delta(G, r_{j\cdot 2cn+k,x}) = \Delta(\overline{C}, r_{j\cdot 2cn+k-1,x}) = \delta < t_j$. Hence, G is a polynomial of degree at most $d - j\cdot 2cn - k$ within distance t_j of $r_{j\cdot 2cn+k,x}$.

For the reverse inclusion, suppose G is a polynomial of degree at most $d - j\cdot 2cn - k$ at distance $\delta < t_j$ of $r_{j\cdot 2cn+k,x}$. Then $\overline{C} := G + Q_{j\cdot 2cn+k-1}(x)Y^{d-j\cdot 2cn-k+1} \in L_{j,k-1,x}$. Since $\mathrm{Coeff}_{Y^{d-j\cdot 2cn-k+1}}(\overline{C}) = Q_{j\cdot 2cn+k-1}(x)$, we get that $G = \overline{C} - Q_{j\cdot 2cn+k-1}(x)Y^{d-j\cdot 2cn-k+1} \in L_{j,k,x}$, as desired.

It remains to show that $Q_{j\cdot 2cn+k}(X) = P_{j\cdot 2cn+k}(X)$. As in the proof of Theorem 4.1, we show that $\Delta(f_{j\cdot 2cn+k}(X), P_{j\cdot 2cn+k}(X)) < \frac{n-j-\sqrt{n}}{2}$, so that the output of FAST-RS-DECODER$(f_{j\cdot 2cn+k}(X), j)$ is $P_{j\cdot 2cn+k}(X)$. Using the first part of the induction we just proved, we get the same $f_{j\cdot 2cn+k}(X)$ as in Algorithm 2. This means we can adopt a nearly identical argument to get to this step:

$$\Delta(f_{j\cdot 2cn+k}(X), P_{j\cdot 2cn+k}(X)) \leq \frac{\Delta(r_{j\cdot 2cn+k}(X,Y), C_{j\cdot 2cn+k}(X,Y))}{n - d + j\cdot 2cn + k}.$$

From here, we get:

$$\begin{aligned}\Delta(f_{j\cdot 2cn+k}(X), P_{j\cdot 2cn+k}(X)) &< \frac{n^2}{2}\left(1-\frac{d}{n}-\frac{1}{\sqrt{n}}\right)\frac{1}{n-d+j\cdot 2cn+k}\\ &= \frac{n}{2}\cdot\frac{n-d-\sqrt{n}}{n-d+j\cdot 2cn+k}\\ &\leq \frac{n}{2}\cdot\frac{n-j\cdot 2cn-k-\sqrt{n}}{n}\\ &= \frac{n-j\cdot 2cn-k-\sqrt{n}}{2}.\end{aligned}$$

Analysis of Runtime of Bivariate Reed-Muller Decoder

We run RS-LIST-DECODER $\frac{d}{2cn}n = \frac{\alpha}{2c}n = \frac{4\alpha}{(1-\alpha)^2}n$ times. Also, we run FAST-RS-DECODER $d=\alpha n$ times. As both of these algorithms run in $O(n\operatorname{polylog} n)$ time, the total runtime of the algorithm is $O(n^2\operatorname{polylog}(n,|\mathbb{F}|))$, after accounting for field operations. As the input is of size n^2, this is near-linear in the size of the input. ◀

6 Near-Linear Time Decoding in the General Case

A more involved variation of the near-linear time decoding algorithm for bivariate Reed-Muller codes can be used to get a near-linear time algorithm for decoding Reed-Muller codes on any number of variables:

▶ **Theorem 6.1.** *Let $\mathbb{F}$ be a finite field and let $S\subseteq\mathbb{F}$ be a nonempty subset of size $|S|=n$. Let $\beta>\frac{1}{2}$. Given a received word $r: S^m\to\mathbb{F}$, there is a $O\left(n^m\cdot\operatorname{polylog}(n,|\mathbb{F}|)\right)$ time algorithm to find the unique polynomial (if it exists) $C\in\mathbb{F}[X_1,\ldots,X_m]$ with $\deg(C)\leq d$ such that*

$$\Delta(r,C) < \frac{n^m}{2}\left(1-\frac{d+(m-1)\beta\sqrt{n}}{n}\right).$$

As part of the algorithm for near linear time Reed-Muller decoding, we will need to decode Reed-Muller codes with uncertainties to various radii less than half their minimum distance. We require the following theorem to do such decodings efficiently.

▶ **Theorem 6.2.** *Let $\mathbb{F}$ be a finite field and let $S\subseteq\mathbb{F}$ be a nonempty subset of size $|S|=n$. Let $\beta>\frac{1}{2}$, and let e be an integer satisfying $0\leq e<n-d-m\beta\sqrt{n}$. Given a received word with uncertainties $r: S^m\to\mathbb{F}\times[0,1]$, there is a $O\left(\frac{n^{m+1}}{e+1}\cdot\operatorname{polylog}(n,|\mathbb{F}|)\right)$ time algorithm to find the unique polynomial (if it exists) $C\in\mathbb{F}[X_1,\ldots,X_m]$ with $\deg(C)\leq d$ such that*

$$\Delta(r,C) < \frac{n^m}{2}\left(1-\frac{d+m\beta\sqrt{n}+e}{n}\right).$$

▶ Remark. The algorithm requires the use of the FAST-RS-DECODER to a radius that is $\beta\sqrt{n}$ less than half the minimum distance. As long as $\beta>\frac{1}{2}$, the FAST-RS-DECODER runs in $O(n\operatorname{polylog} n)$ time.

Proof of Theorem 6.2. The proof is by induction on the number of variables m. The proof of the base case of $m=2$ is similar to the proof of the inductive step and will be handled last. Assume the theorem statement is true for m, and let RM-UNC-DECODER(f,d,s) denote the $O\left(\frac{n^{m+1}}{e+1}\cdot\operatorname{polylog}(n,|\mathbb{F}|)\right)$ time algorithm that finds the unique polynomial (if it

exists) of degree at most d within distance s from f, where $f : S^m \to \mathbb{F} \times [0,1]$ and s can be written as $\frac{n^m}{2}\left(1 - \frac{d+m\beta\sqrt{n}+e}{n}\right)$. We want to show that the theorem statement holds for $m+1$ variables.

The algorithm proceeds as follows: As before, we write $C(\boldsymbol{X}, Y) = \sum_{i=0}^{d} P_i(\boldsymbol{X})Y^{d-i}$, and find the P_i iteratively. In the i-th iteration, decode row $r_{i,\boldsymbol{x}}$, $\boldsymbol{x} \in S^m$ to a degree $d-i$ polynomial within radius $\frac{1}{2}(n-d+i-\beta\sqrt{n}-e)$ to get $D_{i,\boldsymbol{x}}(Y)$. To reduce the number of times we decode, we will instead decode to the larger radius $\frac{1}{2}(n-d+i-\beta\sqrt{n})$ and use this decoding for $e+1$ iterations. Construct the function $f_i : S^m \to \mathbb{F} \times [0,1]$ of (leading coefficient, uncertainty) $= \left(\mathrm{Coeff}_{Y^{d-i}}(D_{i,\boldsymbol{x}}), \frac{\Delta(r_{i,\boldsymbol{x}}, D_{i,\boldsymbol{x}})}{(n-d+i-\beta\sqrt{n}-e)/2}\right)$. Finally, decode $f_i(\boldsymbol{X})$ to a degree i polynomial within radius $\frac{n^m}{2}\left(1 - \frac{i+m\beta\sqrt{n}}{n-d+i}\right)$ to get $Q_i(\boldsymbol{X})$.

Proof of Correctness

We have to show $Q_i(\boldsymbol{X}) = P_i(\boldsymbol{X})$. It is enough to show that

$$\Delta(f_i, P_i) < \frac{n^m}{2}\left(1 - \frac{i+m\beta\sqrt{n}}{n-d+i}\right) < \frac{n^m}{2}\left(1 - \frac{i}{n}\right).$$

Then P_i will be the unique polynomial of degree i within distance $\frac{n^m}{2}\left(1 - \frac{i+m\beta\sqrt{n}}{n-d+i}\right)$ of f_i. Since Q_i is a polynomial of degree i within distance $\frac{n^m}{2}\left(1 - \frac{i+m\beta\sqrt{n}}{n-d+i}\right)$ of f_i, Q_i must be equal to P_i.

When we decode $r_{i,\boldsymbol{x}}$ to radius $\frac{1}{2}(n-d+i-\beta\sqrt{n}-e)$, there are four possibilities:

1. The decoding is unsuccessful. In this case, we set $D_{i,\boldsymbol{x}}$ to be any polynomial of degree $n-d+i$ and set the uncertainty $u_i = 1$. The contribution to $\Delta(f_i, P_i)$ is $\Delta(f_i(\boldsymbol{x}), P_i(\boldsymbol{x})) = 1/2$, which is bounded above by $\frac{1}{2}\frac{\Delta(r_{i,\boldsymbol{x}}, C_{i,\boldsymbol{x}})}{(n-d+i-\beta\sqrt{n}-e)/2}$.
2. The decoding succeeds and is correct. In this case, $D_{i,\boldsymbol{x}} = C_{i,\boldsymbol{x}}$, so $\Delta(f_i(\boldsymbol{x}), P_i(\boldsymbol{x})) = \frac{1}{2}\frac{\Delta(r_{i,\boldsymbol{x}}, C_{i,\boldsymbol{x}})}{(n-d+i-\beta\sqrt{n}-e)/2}$.
3. The decoding succeeds, but is the wrong codeword, whose leading coefficient disagrees with that of the correct codeword. In this case, $D_{i,\boldsymbol{x}} \neq C_{i,\boldsymbol{x}}$, so

$$\begin{aligned}
\Delta(f_i(\boldsymbol{x}), P_i(\boldsymbol{x})) &= 1 - \frac{1}{2}\frac{\Delta(r_{i,\boldsymbol{x}}, D_{i,\boldsymbol{x}})}{(n-d+i-\beta\sqrt{n}-e)/2} \\
&\leq 1 - \frac{(n-d+i) - \Delta(r_{i,\boldsymbol{x}}, C_{i,\boldsymbol{x}})}{(n-d+i-\beta\sqrt{n}-e)} \\
&\leq 1 - \frac{(n-d+i-\beta\sqrt{n}-e) - \Delta(r_{i,\boldsymbol{x}}, C_{i,\boldsymbol{x}})}{(n-d+i-\beta\sqrt{n}-e)} \\
&\leq \frac{\Delta(r_{i,\boldsymbol{x}}, C_{i,\boldsymbol{x}})}{(n-d+i-\beta\sqrt{n}-e)}.
\end{aligned}$$

4. The decoding succeeds, but is the wrong codeword, whose leading coefficient matches that of the correct codeword. As in the previous case, $D_{i,\boldsymbol{x}} \neq C_{i,\boldsymbol{x}}$, and we have:

$$\begin{aligned}
\Delta(f_i(\boldsymbol{x}), P_i(\boldsymbol{x})) &= \frac{1}{2}\frac{\Delta(r_{i,\boldsymbol{x}}, D_{i,\boldsymbol{x}})}{(n-d+i-\beta\sqrt{n}-e)/2} \\
&\leq 1 - \frac{1}{2}\frac{\Delta(r_{i,\boldsymbol{x}}, D_{i,\boldsymbol{x}})}{(n-d+i-\beta\sqrt{n}-e)/2} \\
&\leq \frac{\Delta(r_{i,\boldsymbol{x}}, C_{i,\boldsymbol{x}})}{(n-d+i-\beta\sqrt{n}-e)}.
\end{aligned}$$

Algorithm 4 Decoding Reed-Muller with Uncertainties

Input: $r : S^{m+1} \to \mathbb{F} \times [0,1]$.
for $j = 0, 1, \ldots, \frac{d}{e+1}$ **do**
  Let $t_j = \frac{n-d+j\cdot(e+1)-\beta\sqrt{n}}{2}$.
  Define $r_{j\cdot(e+1)} : S^m \times S \to \mathbb{F}$ by

$$r_{j\cdot(e+1)}(\boldsymbol{X}, Y) = r(\boldsymbol{X}, Y) - \sum_{i=0}^{j\cdot(e+1)-1} Q_i(\boldsymbol{X}) Y^{d-i}.$$

  for $\boldsymbol{x} \in S^m$ **do**
    Define $r_{j\cdot(e+1),\boldsymbol{x}} : S \to \mathbb{F}$ by

$$r_{j\cdot(e+1),\boldsymbol{x}}(Y) = r_{j\cdot(e+1)}(\boldsymbol{x}, Y).$$

    Define $D_{j,0,\boldsymbol{x}}(Y) = \text{FAST-RS-DECODER}(r_{j\cdot(e+1),\boldsymbol{x}}(Y), d - j\cdot(e+1), t_j)$.
    Define $\delta_{\boldsymbol{x}} = \Delta(D_{j,0,\boldsymbol{x}}(Y), r_{j\cdot(e+1),\boldsymbol{x}}(Y))$.
  end for
  for $k = 0, 1, \ldots, e$ **do**
    for $\boldsymbol{x} \in S^m$ **do**
      if $\deg(D_{j,k,\boldsymbol{x}}(Y)) \le d - j\cdot(e+1) - k$ **then**

$$\sigma_{\boldsymbol{x}} \leftarrow \text{Coeff}_{Y^{d-j\cdot(e+1)-k}}(D_{j,k,\boldsymbol{x}}(Y)).$$

      end if
    end for
    Define the weighted function $f_{j\cdot(e+1)+k} : S^m \to \mathbb{F} \times [0,1]$ by

$$f_{j\cdot(e+1)+k}(\boldsymbol{x}) = \left(\sigma_{\boldsymbol{x}}, \min\left\{1, \frac{\delta_{\boldsymbol{x}}}{(n-d+j\cdot(e+1)+k-\beta\sqrt{n}-e)/2}\right\}\right).$$

    Define $Q_{j\cdot(e+1)+k} : S^m \to \mathbb{F}$ by

$$Q_{j\cdot(e+1)+k}(\boldsymbol{X}) = \text{RM-UNC-DECODER}\left(f_{j\cdot(e+1)+k}(\boldsymbol{X}), j\cdot(e+1)+k, \frac{n^m}{2}\left(1 - \frac{j\cdot(e+1)+k+m\beta\sqrt{n}}{n-d+j\cdot(e+1)+k}\right)\right).$$

    for $\boldsymbol{x} \in S^m$ **do**
      Define $D_{j,k+1,\boldsymbol{x}} : S \to \mathbb{F}$ by

$$D_{j,k+1,\boldsymbol{x}} = D_{j,k,\boldsymbol{x}} - Q_{j\cdot(e+1)+k}(\boldsymbol{x}) Y^{d-j\cdot(e+1)-k}.$$

    end for
  end for
end for
Output: $\sum_{i=0}^{d} Q_i(\boldsymbol{X}) Y^{d-i}$.

Putting it all together, we have:

$$\begin{aligned}
\Delta(f_i, P_i) &\leq \sum_{\boldsymbol{x} \in S^m} \frac{\Delta(r_{i,\boldsymbol{x}}, C_{i,\boldsymbol{x}})}{n-d+i-\beta\sqrt{n}-e} \\
&= \frac{\Delta(r_i, C_i)}{n-d+i-\beta\sqrt{n}-e} \\
&= \frac{\Delta(r, C)}{n-d+i-\beta\sqrt{n}-e} \\
&\leq \frac{\frac{n^{m+1}}{2}\left(1-\frac{d+(m+1)\beta\sqrt{n}+e}{n}\right)}{n-d+i-\beta\sqrt{n}-e} \\
&= \frac{n^m}{2}\frac{n-d-(m+1)\beta\sqrt{n}-e}{n-d+i-\beta\sqrt{n}-e} \\
&\leq \frac{n^m}{2}\frac{n-d-m\beta\sqrt{n}}{n-d+i} \\
&= \frac{n^m}{2}\left(1-\frac{i+m\beta\sqrt{n}}{n-d+i}\right).
\end{aligned}$$

Analysis of Runtime

The algorithm can be divided into two parts:

1. Constructing the f_i, $i = 0, \ldots, d$.
2. Decoding the f_i to get the P_i, $i = 0, \ldots, d$.

The dominant contribution to the runtime when constructing f_i comes from all the Reed-Solomon decodings with uncertainties we have to do to get the $D_{i,\boldsymbol{x}}(Y)$. For every $e+1$ iterations, we have to decode each row $x \in S^m$ again. The total number of such decodings is given by $\frac{n}{e+1} \cdot n^m = \frac{n^{m+1}}{e+1}$. Since each Reed-Solomon decoding with uncertainty can be done in $O(n \operatorname{polylog} n)$ time via the FAST-RS-DECODER, we have that the runtime of this part of the algorithm is $O\left(\frac{n^{m+2}}{e+1} \operatorname{polylog} n\right)$.

To understand the runtime of the second part of the algorithm, we will compute the runtime of decoding f_i for some fixed i. Note that decoding f_i is a Reed-Muller decoding with uncertainties problem with m variables. So we will write the decoding radius $\frac{n^m}{2}\left(1-\frac{i+m\beta\sqrt{n}}{n-d+i}\right)$ in the form $\frac{n^m}{2}\left(1-\frac{i+m\beta\sqrt{n}+e_i}{n}\right)$ and apply the induction hypothesis to get a $O\left(\frac{n^{m+1}}{e_i+1} \cdot \operatorname{polylog} n\right)$ runtime. We now need to compute e_i:

$$\begin{aligned}
e_i &= n\frac{i+m\beta\sqrt{n}}{n-d+i} - (i+m\beta\sqrt{n}) \\
&= (i+m\beta\sqrt{n})\left(\frac{n}{n-d+i}-1\right) \\
&= \frac{(i+m\beta\sqrt{n})(d-i)}{n-d+i}.
\end{aligned}$$

The runtime for all $d+1$ iterations from $i = 0, \ldots, d$ is then

$$O\left(\sum_{i=0}^{d} \frac{1}{e_i+1} \cdot n^{m+1} \operatorname{polylog} n\right).$$

It remains to bound $\sum_{i=0}^{d} \frac{1}{e_i+1}$ from above:

$$\begin{aligned}\sum_{i=0}^{d} \frac{1}{e_i+1} &\leq \sum_{i=0}^{d} \min\left(1, \frac{1}{e_i}\right) \\ &\leq 4 + \sum_{i=2}^{d-2} \frac{1}{e_i} \\ &\leq 4 + \int_{1}^{d-1} \frac{n-d+t}{(t+m\beta\sqrt{n})(d-t)} dt.\end{aligned}$$

The last inequality is a simple Riemann sum bound using the fact that the function $\frac{n-d+t}{(t+m\beta\sqrt{n})(d-t)}$ decreases then increases continuously on $[1, d-1]$. Computing the integral is a straightforward partial fraction decomposition:

$$\begin{aligned}\frac{n-d+t}{(t+m\beta\sqrt{n})(d-t)} &= \frac{n}{(t+m\beta\sqrt{n})(d-t)} - \frac{1}{t+m\beta\sqrt{n}} \\ &= \frac{n}{d+m\beta\sqrt{n}}\left(\frac{1}{t+m\beta\sqrt{n}} + \frac{1}{d-t}\right) - \frac{1}{t+m\beta\sqrt{n}} \\ &\leq \frac{1}{\alpha}\left(\frac{1}{t+m\beta\sqrt{n}} + \frac{1}{d-t}\right) - \frac{1}{t+m\beta\sqrt{n}} \\ &= \left(\frac{1}{\alpha} - 1\right)\frac{1}{t+m\beta\sqrt{n}} + \frac{1}{\alpha}\cdot\frac{1}{d-t}\end{aligned}$$

So we have:

$$\begin{aligned}\int_{1}^{d-1} \frac{n-d+t}{(t+m\beta\sqrt{n})(d-t)} dt &\leq \int_{1}^{d-1} \left[\left(\frac{1}{\alpha} - 1\right)\frac{1}{t+m\beta\sqrt{n}} + \frac{1}{\alpha}\cdot\frac{1}{d-t}\right] dt \\ &\leq O\left(\left(\frac{1}{\alpha} - 1\right)\log n + \frac{1}{\alpha}\log n\right) \\ &= O\left(\left(\frac{2}{\alpha} - 1\right)\log n\right) \\ &= O(\log n).\end{aligned}$$

So the runtime for all $d+1$ iterations is:

$$O\left((4 + O(\log n)) \cdot n^{m+1} \operatorname{polylog} n\right) = O(n^{m+1} \operatorname{polylog} n).$$

This means the runtime for both parts of the algorithm is just $O\left(\frac{n^{m+2}}{e+1} \operatorname{polylog} n\right)$.

Base Case

The algorithm for $m = 2$ is almost identical to that for general m, except that we decode $f_i(X)$ to a degree i polynomial within the larger radius $\frac{n}{2}\left(1 - \frac{i+\beta\sqrt{n}}{n}\right)$ to get $Q_i(\boldsymbol{X})$. Note that this radius is still less than half the minimum distance of the Reed-Solomon code of degree i. The correctness of the algorithm follows from the fact that P_i is still the unique polynomial within distance $\frac{n}{2}\left(1 - \frac{i+\beta\sqrt{n}}{n}\right)$ of f_i.

We can again analyze the runtime of the two parts of the algorithm. The runtime for finding the f_i follows the same analysis as before and is $O(\frac{n^3}{e+1} \operatorname{polylog} n)$. For decoding the f_i, we simply call the FAST-RS-DECODER for $d+1$ different values of i. This has a runtime of $O(dn \operatorname{polylog} n) \leq O(n^2 \operatorname{polylog} n)$. So we get a total runtime of $O(\frac{n^3}{e+1} \operatorname{polylog} n)$. ◀

The algorithm for general Reed-Muller decoding follows the same strategy as the algorithm for Reed-Muller decoding with uncertainties to a radius less than half the minimum distance. Recall that to get the f_i in the algorithm, we only needed to Reed-Solomon decode to a radius significantly less than half the minimum distance. We then saved on the number of Reed-Solomon decodings by instead decoding to half the minimum distance and reusing that decoding for many iterations. We now want to Reed-Muller decode to near half the minimum distance. Using the same algorithm doesn't save on enough Reed-Solomon decodings to achieve near linear time. However, when there are no uncertainties in the original received word, we can list decode efficiently to a radius significantly larger than half the minimum distance. We then use the lists for many iterations to generate the f_i before list decoding again.

Proof of Theorem 6.1. In the case where the number of variables is 2, we are in the setting of decoding bivariate Reed-Muller codes to near half the minimum distance, which can be done in near-linear time by Theorem 5.1. Assume now that $m \geq 2$ and that we have a Reed-Muller code in $m+1$ variables.

The decoding algorithm for a $m+1$-variate Reed-Muller code is as follows: In the i-th iteration, list decode row $r_{i,\boldsymbol{x}}$, $\boldsymbol{x} \in S^m$ to obtain a list $L_{i,\boldsymbol{x}}$ of all degree $\leq d-i$ polynomials within radius $\frac{1}{2}(n-d+i+cn)$ along with their distances from $r_{i,\boldsymbol{x}}$, where $c = \frac{(1-\alpha)^2}{8}$. Search the list to get the degree $\leq d-i$ polynomial within distance $\frac{1}{2}(n-d+i)$ from $r_{i,\boldsymbol{x}}$, call it $D_{i,\boldsymbol{x}}(Y)$. We use the lists for cn iterations before list decoding again. Construct function $f_i : S^m \to \mathbb{F} \times [0,1]$ of (leading coefficient, uncertainty) $= \left(\text{Coeff}_{Y^{d-i}}(D_{i,\boldsymbol{x}}), \frac{\Delta(r_{i,\boldsymbol{x}}, D_{i,\boldsymbol{x}})}{(n-d+i)/2}\right)$. Decode $f_i(\boldsymbol{X})$ to a degree i polynomial within radius $\frac{n^m}{2}\left(1 - \frac{i+m\beta\sqrt{n}}{n-d+i}\right)$ to get $Q_i(\boldsymbol{X})$.

Proof of Correctness

As before, we want to show that $Q_i(\boldsymbol{X}) = P_i(\boldsymbol{X})$. It is enough to show

$$\Delta(f_i, P_i) < \frac{n^m}{2}\left(1 - \frac{i+m\beta\sqrt{n}}{n-d+i}\right).$$

We can use a similar analysis of $\Delta(f_i, P_i)$ to the one in Theorem 6.2 to get to the following step:

$$\Delta(f_i, P_i) \leq \frac{\Delta(r, C)}{n-d+i}.$$

So we have:

$$\begin{aligned}\Delta(f_i, P_i) &\leq \frac{\frac{n^{m+1}}{2}\left(1 - \frac{d+m\beta\sqrt{n}}{n}\right)}{n-d+i} \\ &= \frac{n^m}{2}\frac{n-d-m\beta\sqrt{n}}{n-d+i} \\ &= \frac{n^m}{2}\left(1 - \frac{i+m\beta\sqrt{n}}{n-d+i}\right).\end{aligned}$$

Analysis of Runtime

Decoding the f_i over the $d+1$ values of i can be done in $O(n^{m+1}\operatorname{polylog} n)$ following the same runtime analysis from Theorem 6.2. For constructing the f_i, we do $O(n^m)$ Reed-Solomon list decodings taking $O(n \operatorname{polylog} n)$ time each. Within any given list, we need to compute

Algorithm 5 Decoding Reed-Muller

Input: $r : S^{m+1} \to \mathbb{F}$.
Let $c = ((1-\alpha)^2/8)$.
for $j = 0, 1, \ldots, \frac{d}{2cn}$ **do**
  Let $t_j = \frac{n-d+j\cdot 2cn}{2} + cn$.
  Define $r_{j\cdot 2cn} : S^m \times S \to \mathbb{F}$ by

$$r_{j\cdot 2cn}(\boldsymbol{X}, Y) = r(\boldsymbol{X}, Y) - \sum_{i=0}^{j\cdot 2cn-1} Q_i(\boldsymbol{X}) Y^{d-i}.$$

  for $\boldsymbol{x} \in S^m$ **do**
    Define $r_{j\cdot 2cn,\boldsymbol{x}} : S \to \mathbb{F}$ by

$$r_{j\cdot 2cn,\boldsymbol{x}}(Y) = r_{j\cdot 2cn}(\boldsymbol{x}, Y).$$

    Define $L_{j,0,\boldsymbol{x}} = \text{RS-LIST-DECODER}(r_{j\cdot 2cn,\boldsymbol{x}}(Y), d - j\cdot 2cn, t_j)$.
  end for
  for $k = 0, 1, \ldots, 2cn-1$ **do**
    for $\boldsymbol{x} \in S^m$ **do**
      Define $(G_x(Y), \delta_x) \in L_{j,k,x}$ to be the unique codeword (if any) with

$$\delta_x < \frac{n - d + j\cdot 2cn + k}{2}$$

      $\sigma_x \leftarrow \text{Coeff}_{Y^{d-j\cdot 2cn-k}}(G_x)$.
    end for
    Define the weighted function $f_{j\cdot 2cn+k} : S^m \to \mathbb{F} \times [0,1]$ by

$$f_{j\cdot 2cn+k}(\boldsymbol{x}) = \left(\sigma_{\boldsymbol{x}}, \min\left\{1, \frac{\delta_{\boldsymbol{x}}}{(n-d+j\cdot 2cn+k)/2}\right\}\right).$$

    Define $Q_{j\cdot 2cn+k} : S^m \to \mathbb{F}$ by

$$Q_{j\cdot 2cn+k}(\boldsymbol{X}) = \text{RM-UNC-DECODER}\left(f_{j\cdot 2cn+k}(\boldsymbol{X}), j\cdot 2cn + k, \frac{n^{m-1}}{2}\left(1 - \frac{j\cdot 2cn + k + (m-1)\beta\sqrt{n}}{n - d + j\cdot 2cn + k}\right)\right).$$

    for $\boldsymbol{x} \in S^m$ **do**

$$L_{j,k+1,\boldsymbol{x}} \leftarrow \{(C - Q_{j\cdot 2cn+k}(\boldsymbol{x}) Y^{d-j\cdot 2cn-k}, \delta_{C,\boldsymbol{x}}) \mid (C, \delta_{C,\boldsymbol{x}}) \in L_{j,k,\boldsymbol{x}}, \text{Coeff}_{Y^{d-j\cdot 2cn-k}}(C) = Q_{j\cdot 2cn+k}(\boldsymbol{x})\}.$$

    end for
  end for
end for
Output: $\sum_{i=0}^{d} Q_i(\boldsymbol{X}) Y^{d-i}$.

uncertainties for each element of the list. This also takes $O(n \operatorname{polylog} n)$ time for each list. Finally, we update the lists at each iteration by identifying the elements with the correct leading coefficient and taking away their leading terms. Since the list size is constant, and there are $O(n^m)$ lists to update in each iteration, the updating takes $O(n^m d) = O(n^{m+1})$ over $d+1$ iterations. Hence the total runtime is $O(n^{m+1} \operatorname{polylog} n)$ as desired. ◀

7 Open Problems

We conclude with some open problems.

1. The problem of list-decoding multivariate polynomial codes up to the Johnson radius is a very interesting open problem left open by our work. Generalizing our approach seems to require progress on another very interesting open problem, that of list-decoding Reed-Solomon concatenated codes. See [16] for the state of the art on this problem.
2. It would be interesting to understand the relationship between our algorithms and the $m+1$-variate interpolation-based list-decoding algorithm of Sudan [9]. Their decoding radii are incomparable, and perhaps there is some insight into the polynomial method, which is known to face some difficulties in > 2 dimensions, that can be gained here.
3. It would be interesting to see if one can decode multiplicity codes [17] on arbitrary product sets upto half their minimum distance. Here too, we know algorithms that decode upto the minimum distance only in the case when S is very algebraically special (from [18]), or if the degree d is very small compared to $|S|$ (via an $m+1$-variate interpolation algorithm, similar to [9]).

Acknowledgments. We are grateful to Madhu Sudan for introducing this problem to us many years ago.

References

1 D. E. Muller, *Application of boolean algebra to switching circuit design and to error detection*, Electronic Computers, Transactions of the I.R.E. Professional Group on EC-3, 3:6–12, 1954.

2 I. Reed, *A class of multiple-error-correcting codes and the decoding scheme*, Information Theory, Transactions of the I.R.E. Professional Group on 4, 4:38–49, 1954.

3 I. S. Reed and G. Solomon, *Polynomial Codes over Certain Finite Fields*, Journal of the Society for Industrial and Applied Mathematics (SIAM) 8:300–304, 1960.

4 J. L. Massey, *Shift-register synthesis and BCH decoding*, IEEE Transactions on Information Theory, 15:122–127, 1969.

5 F. J. MacWilliams and N. J. A. Sloane, *The theory of error correcting codes*, Elsevier, 1977.

6 V. Guruswami and M. Sudan, *Improved decoding of Reed-Solomon and Algebraic-geometric codes* IEEE Transactions on Information Theory, 45:1757–1767, 1999.

7 R. Pellikaan and X. Wu, *List decoding of q-ary Reed-Muller codes* IEEE Transactions on Information Theory, 50:679–682, 2004.

8 T. Kasami, S. Lin, and W. Peterson, *Polynomial Codes*, IEEE Transactions on Information Theory, 14:807–814, 2006.

9 M. Sudan, *Decoding of Reed Solomon Codes beyond the Error-Correction Bound*, J. Complexity, 13:180–193, 1997.

10 P. Gopalan, V. Guruswami, and P. Raghavendra, *List Decoding Tensor Products and Interleaved Codes*, SIAM J. Comput., 40:1432–1462, 2011.

11 E. Berlekamp, *Bounded distance +1 soft-decision Reed-Solomon decoding*, IEEE Transactions on Information Theory, 42:704–720, 1996.

12 L. R. Welch and E. R. Berlekamp, *Error correction of algebraic block codes*, US Patent Number 4633470, 1986.

13 G. D. Forney, *Generalized Minimum Distance decoding*, IEEE Transactions on Information Theory, 12:125–131, 1966.

14 Y. Cassuto and J. Bruck, *A Combinatorial Bound on the List Size*, Paradise Laboratory Technical report, California Institute of Technology, 2004.

15 M. Alekhnovich, *Linear Diophantine Equations Over Polynomials and Soft Decoding of Reed-Solomon Codes*, IEEE Transactions on Information Theory, 51:2257–2265, 2005.

16 V. Guruswami and M. Sudan, *Decoding concatenated codes using soft information*, Proceedings of the 17th IEEE Annual Conference on Computational Complexity, pp. 122–131, 2002.

17 S. Kopparty, S. Saraf, and S. Yekhanin, *High-rate Codes with Sublinear-time Decoding*, Journal of the ACM, 61, 28:1–20, 2014.

18 S. Kopparty, *List decoding multiplicity codes*, Theory of Computing, 11:149–182, 2015.

A Near-Linear Time Soft Decoding of Reed-Solomon Codes

In this section, we present a near-linear time algorithm to soft decode Reed-Solomon codes to almost half the minimum distance. This result can be used to achieve near-linear time decoding of Reed-Muller codes to almost half the minimum distance.

▶ **Lemma A.1.** *Let $\mathbb{F}$ be a finite field and let $S \subseteq \mathbb{F}$ be a nonempty subset of size $|S| = n$. There is a randomized algorithm FAST-RS-DECODER(r, d) that given a received word with uncertainties $r : S \to \mathbb{F} \times [0, 1]$, finds the unique polynomial (if it exists) $C \in \mathbb{F}[X]$ satisfying $\deg(C) \leq d$ and $\Delta(r, C) < \frac{n-d-\sqrt{n}}{2}$ with probability $3/4$ in time $O(n \operatorname{polylog}(n))$.*

Proof. The near-linear time algorithm for FAST-RS-DECODER(r, d) is based on Forney's generalized minimum distance decoding of concatenated codes.

Given a received word $r : S \to \mathbb{F} \times [0, 1]$, suppose there is a polynomial f of degree at most d such that $\Delta(f, r) < \frac{n-d-\sqrt{n}}{2}$. Let $S = \{\alpha_1, \alpha_2, \ldots, \alpha_n\}$, and write $r(\alpha_i) = (\beta_i, u_i), i \in [n]$. We may view r as a set of n points (α_i, β_i) with uncertainties u_i. The general idea of the algorithm is to erase the i-th point with probability u_i, and perform errors and erasures decoding of the resulting Reed-Solomon code. We denote the errors and erasures Reed-Solomon decoder by EE-DECODER(r', d), which takes a received word $r' : S \to \mathbb{F} \times [0, 1] \cup \{?\}$ and a degree d and returns the polynomial of degree at most d that is within $\frac{n-d}{2}$ of r'.

Algorithm 6 Fast Reed-Solomon Decoding with Uncertainties

Input: $r : S \to \mathbb{F} \times [0, 1]$.
for $i = 1, 2, \ldots, n$ **do**
  $p_i \leftarrow \text{RANDOM}([0, 1])$.
  Define $r' : S \to (\mathbb{F} \cup \{?\})$ by

$$r'(\alpha_i) = \begin{cases} \beta_i & \text{if } p_i \leq u_i \\ ? & \text{if } p_i > u_i \end{cases}.$$

end for
$g \leftarrow$ EE-DECODER(r', d).
Output: g.

We say that a point is an *erasure* if it is erased by the algorithm. We say that a point (α_i, β_i) is an *error* if (α, β) is not an erasure and $f(\alpha_i) \neq \beta_i$. Let E be the number of errors, and let F be the number of erasures. As the resulting $n - F$ points form a Reed-Solomon code of block length $n - F$ and degree d, the algorithm outputs f as long as

$$2E + F < n - d.$$

We will use Chebyshev's inequality to show that $2E + F < n - d$ with probability at least $\frac{3}{4}$. To help us compute the expectation and variance of $2E + F$, we write E and F as a sum of indicator random variables. Let $A = \{i \in [n] | f(\alpha_i) = \beta_i\}$ be the set of agreeing indices, and let $D = \{i \in [n] | f(\alpha_i) \neq \beta_i\}$ be the set of disagreeing indices. Let $T = \{i \in [n] | (\alpha_i, \beta_i) \text{ is erased}\}$ be the set of erasure indices.

Then we can write

$$E = \sum_{i \in D} 1_{i \notin T}$$
$$F = \sum_{i \in [n]} 1_{i \in T}.$$

We then can show $\mathbb{E}[2E + F]$ is less than $n - d$ by a significant amount $\sqrt{n}$:

$$\begin{aligned}
\mathbb{E}[2E + F] &= 2 \sum_{i \in D} (1 - u_i) + \sum_{i \in [n]} u_i \\
&= 2 \sum_{i \in D} (1 - u_i) + \sum_{i \in D} u_i + \sum_{i \in A} u_i \\
&= 2 \left(\sum_{i \in D} \left(1 - \frac{u_i}{2}\right) + \sum_{i \in A} \frac{u_i}{2} \right) \\
&= 2\Delta(f, r) \\
&< n - d - \sqrt{n}.
\end{aligned}$$

Finally, we show that $\mathrm{Var}(2E + F)$ is small:

$$\begin{aligned}
&\mathrm{Var}(2E + F) \\
&= 4\mathrm{Var}(E) + 4\mathrm{Cov}(E, F) + \mathrm{Var}(F) \\
&= 4 \sum_{i \in D} u_i(1 - u_i) + 4 \left(\mathbb{E} \left[\sum_{i \in D} \sum_{j \in [n]} 1_{i \notin T \cap j \in T} \right] - \sum_{i \in D} (1 - u_i) \sum_{j \in [n]} u_j \right) + \sum_{i \in [n]} u_i(1 - u_i) \\
&= 4 \sum_{i \in D} u_i(1 - u_i) + 4 \left(\mathbb{E} \left[\sum_{i \in D} \sum_{j \neq i} (1 - u_i)u_j \right] - \sum_{i \in D} \sum_{j \in [n]} (1 - u_i)u_j \right) + \sum_{i \in [n]} u_i(1 - u_i) \\
&= 4 \sum_{i \in D} u_i(1 - u_i) - 4 \sum_{i \in D} u_i(1 - u_i) + \sum_{i \in [n]} u_i(1 - u_i) \\
&= \sum_{i \in [n]} u_i(1 - u_i) \\
&\leq \frac{n}{4}.
\end{aligned}$$

By Chebyshev's inequality, $\Pr(2E + F \geq n - d) \leq \frac{1}{4}$. Hence we have $\Pr(2E + F < n - d) \geq \frac{3}{4}$. That is, with probability at least $\frac{3}{4}$, the algorithm outputs f.

We now analyze the runtime of our fast Reed-Solomon decoder. The erasures can be done in $O(n)$ time. Also, as the EE-DECODER is essentially a Reed-Solomon decoder to half the minimum distance, it runs in time $O(n \operatorname{polylog} n)$ [11, 12]. This gives a total runtime of $O(n \operatorname{polylog} n)$. ◀

Note that by running the algorithm $\log n$ times, we get that with probability at least $1 - (1/4)^{\log n} = 1 - 1/n^{\log 4}$, we still find f in $O(n \operatorname{polylog} n)$ time.

Lower Bounds for Constant Query Affine-Invariant LCCs and LTCs

Arnab Bhattacharyya[*1] **and Sivakanth Gopi**[†2]

1 **Department of Computer Science and Automation, Indian Institute of Science, Bangalore, India**
arnabb@csa.iisc.ernet.in

2 **Department of Computer Science, Princeton University, Princeton, USA**
sgopi@cs.princeton.edu

Abstract

Affine-invariant codes are codes whose coordinates form a vector space over a finite field and which are invariant under affine transformations of the coordinate space. They form a natural, well-studied class of codes; they include popular codes such as Reed-Muller and Reed-Solomon. A particularly appealing feature of affine-invariant codes is that they seem well-suited to admit local correctors and testers.

In this work, we give lower bounds on the length of locally correctable and locally testable affine-invariant codes with constant query complexity. We show that if a code $\mathcal{C} \subset \Sigma^{\mathbb{K}^n}$ is an r-query locally correctable code (LCC), where $\mathbb{K}$ is a finite field and Σ is a finite alphabet, then the number of codewords in $\mathcal{C}$ is at most $\exp(O_{\mathbb{K},r,|\Sigma|}(n^{r-1}))$. Also, we show that if $\mathcal{C} \subset \Sigma^{\mathbb{K}^n}$ is an r-query locally testable code (LTC), then the number of codewords in $\mathcal{C}$ is at most $\exp(O_{\mathbb{K},r,|\Sigma|}(n^{r-2}))$. The dependence on n in these bounds is tight for constant-query LCCs/LTCs, since Guo, Kopparty and Sudan (ITCS'13) construct affine-invariant codes via lifting that have the same asymptotic tradeoffs. Note that our result holds for non-linear codes, whereas previously, Ben-Sasson and Sudan (RANDOM'11) assumed linearity to derive similar results.

Our analysis uses higher-order Fourier analysis. In particular, we show that the codewords corresponding to an affine-invariant LCC/LTC must be far from each other with respect to Gowers norm of an appropriate order. This then allows us to bound the number of codewords, using known decomposition theorems which approximate any bounded function in terms of a finite number of low-degree non-classical polynomials, upto a small error in the Gowers norm.

1998 ACM Subject Classification E.4 Coding and Information Theory

Keywords and phrases Locally correctable code, Locally testable code, Affine Invariance, Gowers uniformity norm

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.12

1 Introduction

Error-correcting codes which admit local algorithms are of significant interest in theoretical computer science. A code is called a locally correctable code (LCC) if there is a randomized algorithm that, given an index i and a received word w close to a codeword c in Hamming distance, outputs c_i by querying only a few positions of w. A code is called a locally testable

* Supported in part by DSTO1358 Ramanujan Fellowship.
† Partially supported by NSF grants CCF-1523816 and CCF-1217416.

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 12; pp. 12:1–12:17

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

code (LTC) if there is a randomized algorithm that, given a received word w, determines whether w is in the code or whether w is far in Hamming distance from every codeword, based on queries to a small number of locations of w. The number of positions of the received word queried is called the query complexity of the LCC or LTC.

The notions of local correctability and local testability have a long history in computer science by now. Also called "self-correction", the idea of local correction originated in works by Lipton [30] and by Blum and Kannan [13] on program checkers. LCCs are closely related to locally decodable codes (LDCs), where the goal is to recover a symbol of the underlying message when given a corrupted codeword, using a small number of queries [27]. LDCs and LCCs have found applications in private information retrieval schemes [15, 4] and derandomization [32]. See [39] for a detailed survey on LDCs and LCCs. Research on LTCs implicitly started with Blum, Luby, and Rubinfeld's seminal discovery [14] that the Hadamard code is an LTC with query complexity 3; they were first formally defined by Goldreich and Sudan in [20]. LTCs have been used (implicitly and explicitly) in many contexts, most notably in the construction of PCP's [2, 1, 16].

In spite of the wide interest in them, some basic questions about LCCs and LTCs remain unanswered. We restrict ourselves throughout to the setting where the query complexity is a constant (independent of the length of the code) and consider the tradeoff between query complexity and code length. The current best constant-query LCCs have exponential length, while the current best constant-query LTCs have near-linear length but they are quite complicated [7, 16, 31, 36]. Getting subexponential length LCCs or linear length LTCs with constant query complexity are major open problems in the area.

Intuitively, for LCCs and LTCs with constant query complexity, there must be a lot of redundancy in the code, since every symbol of the codeword must satisfy local constraints with most other symbols in the codeword. A systematic way to generate redundancy is to make sure that the code has a large group of *invariances*[1]. Formally, given a code $\mathcal{C} \subset \Sigma^N$ of length N over alphabet Σ, a codeword $c \in \mathcal{C}$ can be naturally viewed as a function $c : [N] \to \Sigma$. Then, we say that $\mathcal{C}$ is invariant under a set[2] $G \subset \{[N] \to [N]\}$ if for every $\pi \in G$ and codeword $c \in \mathcal{C}$, $c \circ \pi$ also describes a codeword $c' \in \mathcal{C}$. Now, the key observation is that if for every codeword $c \in \mathcal{C}$, if there is a constraint among $c(i_1), \ldots, c(i_k)$ for some $i_1, \ldots, i_k \in [N]$, then for every $c \in \mathcal{C}$, there must also be a constraint among $c(\pi(i_1)), \ldots, c(\pi(i_k))$ for any π in the invariance set G, since $c \circ \pi$ is itself another codeword. Hence if G is large, the presence of one local constraint immediately implies presence of many and suggests the possibility of local algorithms for the code. This connection between invariance and correctability/testability was first explicitly examined by Kaufman and Sudan [28]. One is then motivated to understand more clearly the possibilities and limitations of local correctors/testers for codes possessing natural symmetries.

We focus on affine-invariant codes, for which the domain $[N]$ is an n-dimensional vector space $\mathbb{K}^n$ over a finite field $\mathbb{K}$ and the code $\mathcal{C} \subset \{\mathbb{K}^n \to \Sigma\}$ is invariant under affine transformations $A : \mathbb{K}^n \to \mathbb{K}^n$. Affine invariance is a very natural symmetry for "algebraic codes" and has long been studied in coding theory [26]. The study of affine-invariant LCCs and LTCs was initiated in [28] and has been investigated in several follow-up works [8, 24, 5, 25]. The hope is that because affine-invariant codes have a large group of invariance and, at the same time, are conducive to non-trivial algebraic constructions, they may contain

[1] A quite different way to generate redundancy is through *tensoring*; see [6]. Invariances and tensoring are essentially the only two "generic" reasons known to cause local correctability/testability.

[2] $\{A \to B\}$ and B^A denote the set of all functions from A to B.

a code that improves current constructions of LCCs or LTCs.

The current best parameters for constant-query affine-invariant LCCs and LTCs are achieved by the lifted codes of Guo, Kopparty and Sudan [23]. They construct an affine-invariant code $\mathcal{F} \subset \{\mathbb{F}_{2^\ell}^n \to \mathbb{F}_2\}$ with $\exp(\Theta(n^{r-2}))$ codewords that is an $(r-1)$-query LCC and an r-query LTC, where $r = 2^\ell$. The $\Theta(\cdot)$ notation hides factors that depend on r but not n. For LCCs, the same asymptotic tradeoff between query complexity and code length is achieved by the Reed-Muller code. For every $r \geq 2$, the Reed-Muller code of order $r-1$ (i.e., polynomials over $\mathbb{F}_q$ on n variables of total degree $\leq r-1$ with $q > r$) is an affine-invariant r-query LCC with $\exp(\Theta(n^{r-1}))$ codewords. In fact, even if we drop the affine-invariance requirement, Reed-Muller codes and the construction of [23] achieve the best known codeword length for constant query LCCs[3].

In this work, we show that the parameters for the lifted codes of [23] are, in fact, tight for affine-invariant LCCs/LTCs in $\{\mathbb{K}^n \to \Sigma\}$ for any fixed finite field $\mathbb{K}$ and any fixed finite alphabet Σ.

▶ **Theorem 1.1** (Main Result, informal).

(i) *Let $\mathcal{C} \subset \{\mathbb{K}^n \to \Sigma\}$ be an r-query affine-invariant LCC. Then $|\mathcal{C}| \leq \exp\left(O_{\mathbb{K},r,|\Sigma|}(n^{r-1})\right)$.*

(ii) *Let $\mathcal{C} \subset \{\mathbb{K}^n \to \Sigma\}$ be an r-query affine-invariant LTC. Then $|\mathcal{C}| \leq \exp\left(O_{\mathbb{K},r,|\Sigma|}(n^{r-2})\right)$.*

1.1 Related Work

Ben-Sasson and Sudan in [8] obtained a similar result as Theorem 1.1, when the code is assumed to be linear, i.e., when the codewords form a vector space. They showed that if $\mathcal{C} \subset \{\mathbb{K}^n \to \mathbb{F}\}$ is an $(r-1)$-query locally correctable or r-query locally testable *linear*, affine-invariant code, where $\mathbb{K}$ and $\mathbb{F}$ are finite fields of characteristic $p > 0$ with $\mathbb{K}$ an extension of $\mathbb{F}$, then the dimension of $\mathcal{C}$ as a vector space over $\mathbb{F}$ is at most $(n \log_p |\mathbb{K}|)^{r-2}$. When $\mathbb{K}$ is fixed (as in [23]'s construction of constant query LCCs/LTCs), the result of [8] is a very special case of our Theorem 1.1. On the other hand, [8]'s result also applies when the size of $\mathbb{K}$ is growing (as long as $\mathbb{K}$ extends $\mathbb{F}$), whereas ours does not.

There are several works which study lower bounds for constant query LCCs [27, 19, 18, 29, 3, 10, 38, 17]. For general (non-affine-invariant) LCCs, tight lower bounds are known only for 2-query LCCs. Kerendis and deWolf [29] prove that if $\mathcal{C} \subset \{\{0,1\}^n \to \Sigma\}$ is a 2-query LCC[4], then $|\mathcal{C}| \leq \exp(O(n|\Sigma|^5))$. This is tight for constant Σ and achieved by the Hadamard code. For r-query LCCs where $r > 2$, the lower bounds known are much weaker. The best known bounds, due to [29, 37], show that if $\mathcal{C} \subset \{\{0,1\}^n \to \{0,1\}\}$ is an r-query LCC, then

$$|\mathcal{C}| \leq \exp\left(2^{n/(1+1/(\lceil r/2 \rceil+1))+o(n)}\right).$$

Higher-order Fourier analysis was applied to other problems in coding theory in [12, 35].

1.2 Proof Overview

Our arguments are based on standard techniques from higher-order Fourier analysis [33], but they are new in this context. We show that if an affine-invariant code is an r-query LCC,

[3] In contrast, there exist non-affine-invariant LTCs of constant query complexity and inverse polylogarithmic rate. This corresponds to an LTC with $\exp(N/\text{polylog}(N))$ codewords, where N is the code length, while the affine-invariant LTC of [23] and Reed-Muller codes have $\exp(\text{polylog}(N))$ codewords for constant query complexity.

[4] Their lower bound also holds for the weaker notion of locally decodable dode (LDC).

then its codewords are far from each other in the U^r-norm, the *Gowers norm of order* r. Similarly, we show that the codewords of an affine-invariant r-query LTC are far from each other in the U^{r-1}-norm. Therefore, we can upper bound the number of LCC/LTC codewords in terms of the size of a net that is fine enough with respect to the Gowers norm of an appropriate order. We bound the size of such a net by explicitly constructing one using a standard decomposition theorem (analogous to Szemerédi's regularity lemma): any bounded function $f : \mathbb{K}^n \to \mathbb{C}$ can be approximated, upto a small error in the Gowers norm, by a composition of a bounded number of low-degree non-classical polynomials [34].

The way we argue that two codewords f and g of an r-query LCC are far in the Gowers norm is that if $\|f - g\|_{U^r} < \epsilon$, then for small enough ϵ (with respect to r, $|\Sigma|$ and correctness probability), the local corrector when applied to f can act as if it is applied to g. The argument is, briefly, as follows. On the one hand, the codewords f and g must be far in Hamming distance, because the definition of LCC implies that there is a unique codeword close to any string. So, with constant probability over choice of $y \in \mathbb{K}^n$, the local corrector's guess for $f(y)$ must differ from $g(y)$. On the other hand, we can lower bound by a constant the probability of the event that the corrector outputs $g(y)$ when it queries coordinates of f, because f and g are close in the $\|\cdot\|_{U^r}$ norm. This last calculation uses the affine invariance of the code and the *generalized von Neumann inequality*, which bounds by $\|f_0\|_{U^k}$ the expectation over $z_1, \ldots, z_m \in \mathbb{K}^n$ of the product $\prod_{i=0}^{k} f_i(\mathcal{L}_i(z_1, \ldots, z_m))$, where the $\mathcal{L}_i$'s are arbitrary linear forms so that no two are linearly dependent and $f_i : \mathbb{K}^n \to \mathbb{C}$ are arbitrary functions with $|f_i| \leq 1$.

The argument for r-query LTCs is similar. Suppose f and g are close in the $\|\cdot\|_{U^{r-1}}$ norm. Consider the random function H such that for every x independently, $H(x)$ equals $f(x)$ with probability $1/2$ and $g(x)$ with probability $1/2$. H itself is far from a codeword with high probability. But we show that since the local tester accepts f, it will also accept $H \circ \ell$ for a random invertible affine map $\ell : \mathbb{K}^n \to \mathbb{K}^n$ with good probability. This implies that with good probability, $H \circ \ell$ is close to a codeword and by affine-invariance, H itself is close to a codeword which gives a contradiction. To draw this conclusion, we again use the generalized von Neumann inequality as well as a hybrid argument.

Organization

Section 2 contains preliminaries that lay the foundations of our analysis. Section 3 proves the first part of our main result about LCCs, while Section 4 proves the second part about LTCs.

2 Preliminaries

2.1 Error-correcting codes

Let $\mathcal{X}$ be a finite set called the set of coordinates and Σ be an other finite set called the alphabet. Let $\Sigma^{\mathcal{X}}$ denote the set of all functions from $\mathcal{X} \to \Sigma$. A subset $\mathcal{C} \subset \Sigma^{\mathcal{X}}$ is called a code and its elements are called *codewords*.

▶ **Definition 2.1** (Hamming distance). Given $f, g \in \Sigma^{\mathcal{X}}$, we define the normalized Hamming distance between f and g is defined as $\Delta(f, g) := \Pr_{x \in \mathcal{X}}[f(x) \neq g(x)]$ where x is uniformly chosen from $\mathcal{X}$. For a code $\mathcal{C} \subset \Sigma^{\mathcal{X}}$, we define the minimum distance of $\mathcal{C}$ as $\min_{f,g \in \mathcal{C}, f \neq g} \Delta(f, g)$.

Let $\blacktriangle_\Sigma = \{q : \Sigma \to \mathbb{R}_{\geq 0} : \sum_{i \in \Sigma} q(i) = 1\}$ denote the probability simplex on Σ. We embed Σ into $\blacktriangle_\Sigma$ by sending $i \in \Sigma$ to e_i which is the i^{th} coordinate vector in $\mathbb{R}^\Sigma$. This also lets

us extend functions $f : \mathcal{X} \to \Sigma$ to $\hat{f} : \mathcal{X} \to \blacktriangle_\Sigma$ using the embedding. We call $\hat{f}$ the simplex extension of f. Now given $f, g \in \Sigma^{\mathcal{X}}$, we can write the Hamming distance between them as

$$\Delta(f,g) = 1 - \Pr_{x \in \mathcal{X}}[f(x) = g(x)] = 1 - \mathbb{E}_{x \in \mathcal{X}} \langle \hat{f}, \hat{g} \rangle$$

where $\langle \cdot, \cdot \rangle$ is the standard inner product in $\mathbb{R}^\Sigma$.

▶ **Definition 2.2** (Affine invariance)**.** Let $\mathcal{X}$ be a finite dimensional vector space over some finite field $\mathbb{K}$, then $\mathcal{C} \subset \Sigma^{\mathcal{X}}$ is called affine invariant if for every $f \in \mathcal{C}$ and every invertible affine map $\ell : \mathcal{X} \to \mathcal{X}$, $f \circ \ell \in \mathcal{C}$.

Locally correctable and testable codes are defined formally in Sections 3 and 4 respectively.

2.2 Higher order Fourier analysis

Fix a finite field $\mathbb{F}_p$ of prime order p, and let $\mathbb{K} = \mathbb{F}_q$ where $q = p^t$ for a positive integer t. $\mathbb{K}$ is then a vector space of dimension t over $\mathbb{F}_p$. We denote by $\mathrm{Tr} : \mathbb{K} \to \mathbb{F}_p$ the *trace function*:

$$\mathrm{Tr}(x) = x + x^p + x^{p^2} + \cdots + x^{p^{t-1}}.$$

Also, we use $|\cdot|$ to denote the obvious map from $\mathbb{F}_p$ to $\{0, 1, \ldots, p-1\}$.

Given functions $f, g : \mathbb{K}^n \to \mathbb{C}$, we define their inner product as $\langle f, g \rangle = \mathbb{E}_x[\overline{f(x)} g(x)]$ where x is chosen uniformly from $\mathbb{K}^n$. We define $\|\cdot\|_p$-norm on such functions as $\|f\|_p = \mathbb{E}_x[|f(x)|^p]^{1/p}$. We say a function $f : \mathbb{K}^n \to \mathbb{C}$ is *bounded* if $|f| \leq 1$. Let $\mathbb{T}$ denote the circle group $\mathbb{R}/\mathbb{Z}$ and $e : \mathbb{T} \to \mathbb{C}$ be the map given by $e(x) = \exp(2\pi i x)$.

▶ **Definition 2.3** (Non-classical Polynomials)**.** A *non-classical polynomial of degree* $< d$ is a function $f : \mathbb{K}^n \to \mathbb{T}$ if

$$\forall h_1, h_2 \cdots, h_d \in \mathbb{K}^n \quad D_{h_1} D_{h_2} \cdots D_{h_d} f = 0$$

where D_h is the difference operator defined as $D_h f(x) = f(x+h) - f(x)$. For such an f, the function $e(f)$ is called a *non-classical phase polynomial of degree* $< d$.

Let $\alpha_1, \cdots, \alpha_t \in \mathbb{K}$ be a basis for $\mathbb{K}$ when viewed as a vector space over $\mathbb{F}_p$. It is known [34, 9] that non-classical polynomials of degree $\leq d$ are exactly those functions $P : \mathbb{K}^n \to \mathbb{T}$ which have the following form:

$$P(x_1, \ldots, x_n) = \theta + \sum_{k \geq 0} \sum_{\substack{0 \leq d_{i,j} < p \ \forall i \in [n], j \in [t]; \\ 0 < \sum_{i=1}^n \sum_{j=1}^t d_{i,j} \leq d - k(p-1)}} \frac{c_{d_{1,1},\ldots,d_{n,t},k} \prod_{i=1}^n \prod_{j=1}^t |\mathrm{Tr}(\alpha_j x_i)|^{d_{i,j}}}{p^{k+1}} \pmod 1 \tag{1}$$

for some $c_{d_{1,1},\ldots,d_{n,t},k} \in \{0, 1, \cdots, p-1\}$ and $\theta \in \mathbb{T}$. Next, we define the Gowers norm for arbitrary functions $f : \mathbb{K}^n \to \mathbb{C}$.

▶ **Definition 2.4** (Gowers uniformity norm [21])**.** For a function $f : \mathbb{K}^n \to \mathbb{C}$, the *Gowers norm of order* r, denoted by $\|\cdot\|_{U^r}$, is defined as

$$\|f\|_{U^r} = (\mathbb{E}_{x, h_1, \cdots, h_r \in \mathbb{K}^n}[\Delta_{h_1} \Delta_{h_2} \cdots \Delta_{h_r} f(x)])^{1/2^r}$$

where Δ_h is the multiplicative difference operator defined as $\Delta_h f(x) = f(x+h)\overline{f(x)}$.

The Gowers norm is an actual norm when $r \geq 2$. It also satisfies a useful monotonicity property: for any function $f : \mathbb{K}^n \to \mathbb{C}$,

$$|\mathbb{E}[f(x)]| = \|f\|_{U^1} \leq \|f\|_{U^2} \leq \cdots \leq \|f\|_{U^r} \leq \cdots \leq \|f\|_\infty.$$

See [33] for more on Gowers norm. Observe that if $f : \mathbb{K}^n \to \mathbb{C}$ is a non-classical phase polynomial of degree $< r$ then $\|f\|_{U^r} = 1$. The inverse Gowers theorem is a partial converse to this. It shows that the Gowers norm of order r of a function is in direct correspondence with its correlation with non-classical phase polynomials of degree $< r$. In particular:

▶ **Lemma 2.5** (Inverse Gowers theorem [34]). *For any bounded*[5] $f : \mathbb{K}^n \to \mathbb{C}$, *if* $\|f\|_{U^r} > \delta$ *then there exists a non-classical polynomial* P *of degree* $< r$ *such that*

$$|\langle f, e(P)\rangle| \geq c(\delta, \mathbb{K}, r)$$

where $c(\delta, \mathbb{K}, r)$ *is a constant depending only on* $\delta, \mathbb{K}, r$.

A linear form on m variables is a vector $\mathcal{L} = (w_1, \cdots, w_m) \in \mathbb{K}^m$ that is interpreted as a function $\mathcal{L} : (\mathbb{K}^n)^m \to \mathbb{K}^n$ via the map $(x_1, \cdots, x_m) \mapsto \sum_{i=1}^m w_i x_i$. A key reason that the Gowers norm is useful in applications is that if a function has small Gowers norm of the appropriate order, then it behaves pseudorandomly in a certain way with respect to linear forms.

▶ **Lemma 2.6** (Generalized von Neumann inequality (Exercise 1.3.23 in [33])). *Let* $f_0, f_1, f_2, \cdots, f_k : \mathbb{K}^n \to \mathbb{C}$ *be bounded functions and let* $\mathcal{L} = \{\mathcal{L}_0, \mathcal{L}_1, \cdots, \mathcal{L}_k\}$ *be a system of* $k+1$ *linear forms in* m *variables such that no form is a multiple of another. Then*

$$|\mathbb{E}_{z_1,\cdots,z_m \in \mathbb{K}^n}[\prod_{i=0}^{k} f_i(\mathcal{L}_i(z_1, \cdots, z_m))]| \leq \min_{0 \leq i \leq k} \|f_i\|_{U^k}.$$

See Appendix A for proof.

2.3 A net for Gowers norm

The goal of this section is to establish the following claim.

▶ **Theorem 2.7** (ϵ-net for U^r norm). *The metric induced by the* $\|\cdot\|_{U^r}$ *norm on the space of all bounded functions* $\{f : \mathbb{K}^n \to \mathbb{C}\}$ *has an* ϵ*-net of size* $\exp(O_{\epsilon,\mathbb{K},r}(n^{r-1}))$.

For the proof, we need the following definitions.

▶ **Definition 2.8** (Polynomial factors). A polynomial factor $\mathcal{B}$ is a sequence of non-classical polynomials $P_1, ..., P_k : \mathbb{K}^n \to \mathbb{T}$. We also identify it with the function $\mathcal{B} : \mathbb{K}^n \to \mathbb{T}^k$ mapping $x \mapsto (P_1(x), ..., P_k(x))$. The partition induced by $\mathcal{B}$ is the partition of $\mathbb{K}^n$ given by $\{\mathcal{B}^{-1}(y) : y \in \mathbb{T}^k\}$. The complexity of $\mathcal{B}$ is the number of defining polynomials, $|\mathcal{B}| = k$. The degree of $\mathcal{B}$ is the maximum degree among its defining polynomials $P_1, \cdots, P_k$. A function $f : \mathbb{K}^n \to \mathbb{C}$ is called $\mathcal{B}$-measurable if it is constant in each cell of the partition induced by $\mathcal{B}$ or equivalently f can be written as a $\tau(P_1, \cdots, P_k)$ for some function $\tau : \mathbb{T}^k \to \mathbb{C}$.

[5] Note that bounded means $|f| \leq 1$.

▶ **Definition 2.9** (Conditional expectations)**.** Given a polynomial factor $\mathcal{B}$, the conditional expectation of $f : \mathbb{K}^n \to \mathbb{C}$ over $\mathcal{B}$, denoted by $\mathbb{E}[f|\mathcal{B}]$, is the $\mathcal{B}$-measurable function defined by

$$\mathbb{E}[f|\mathcal{B}](x) = \mathbb{E}_{y \in \mathcal{B}^{-1}(\mathcal{B}(x))}[f(y)].$$

▶ **Definition 2.10** (Factor refinement)**.** Given two polynomial factors $\mathcal{B}, \mathcal{B}'$, we say $\mathcal{B}'$ is a refinement of $\mathcal{B}$, denoted by $\mathcal{B}' \preceq \mathcal{B}$, if every cell in the partition induced by $\mathcal{B}'$ is contained in some cell in the partition induced by $\mathcal{B}$.

The definition of refinement immediately implies:

▶ **Lemma 2.11** (Pythagoras theorem)**.** *Let $\mathcal{B}, \mathcal{B}'$ be polynomial factors such that $\mathcal{B}' \preceq \mathcal{B}$, then for any function $f : \mathbb{K}^n \to \mathbb{C}$,*

$$\|\mathbb{E}[f|\mathcal{B}']\|_2^2 = \|\mathbb{E}[f|\mathcal{B}]\|_2^2 + \|\mathbb{E}[f|\mathcal{B}'] - \mathbb{E}[f|\mathcal{B}]\|_2^2.$$

The next claim shows that any bounded function is "close" to being measurable by a polynomial factor of bounded complexity. Precisely:

▶ **Lemma 2.12** (Decomposition Theorem)**.** *Any bounded $f : \mathbb{K}^n \to \mathbb{C}$ can be approximated in $\|\cdot\|_{U^r}$ by a function of a small number of degree $< r$ non-classical polynomials i.e. for any $\epsilon > 0$, there exists non-classical polynomials $P_1, P_2, \cdots, P_k$ of degree $< r$ with $P_i(\bar{0}) = 0$ $\forall i$ and a bounded function $\tau : \mathbb{T}^k \to \mathbb{C}$ such that*

$$\|f - \tau(P_1, P_2, \cdots, P_k)\|_{U^r} \leq \epsilon$$

where $k = k(\epsilon, \mathbb{K}, r)$ is a constant depending only on $\epsilon, \mathbb{K}, r$.

Proof. The proof is similar to the proof of the Quadratic Koopman-von Neumann decompostion which is Prop 3.7 in [22] but using the full Inverse Gowers Theorem (Lemma 2.5) and similar claims are implicit elsewhere, but for completeness, we give the proof.

The main idea is to approximate the function f using its conditional expectation over a suitable polynomial factor $\mathcal{B}$ of degree $< r$. We will start with the trivial factor $\mathcal{B}_0 = (1)$ and iteratively construct more refined partitions $\mathcal{B}_i \preceq \mathcal{B}_{i-1}$ until we find a factor $\mathcal{B}_k$ which satisfies $\|f - \mathbb{E}[f|\mathcal{B}_k]\|_{U^r} \leq \epsilon$. To bound the number of iterations needed to achieve this, we will show that the energy $\|\mathbb{E}[f|\mathcal{B}_i]\|_2^2$ which is bounded above by 1, increases by a fixed constant in every step.
Suppose that after step $i-1$, we still have $\|f - \mathbb{E}[f|\mathcal{B}_{i-1}]\|_{U^r} > \epsilon$. Let $g = f - \mathbb{E}[f|\mathcal{B}_{i-1}]$, then by the inverse Gowers theorem (Lemma 2.5), we have some non-classical polynomial P_i of degree $< r$ such that $|\langle g, e(P_i)\rangle| \geq \kappa = c(\epsilon, p, r)$. We can assume that $P_i(\bar{0}) = 0$. Refine the factor $\mathcal{B}_{i-1}$ by adding the polynomial P_i to obtain $\mathcal{B}_i \preceq \mathcal{B}_{i-1}$. Now consider the energy increment,

$$\|\mathbb{E}[f|\mathcal{B}_i]\|_2^2 - \|\mathbb{E}[f|\mathcal{B}_{i-1}]\|_2^2 = \|\mathbb{E}[f|\mathcal{B}_i] - \mathbb{E}[f|\mathcal{B}_{i-1}]\|_2^2 = \|\mathbb{E}[g|\mathcal{B}_i]\|_2^2$$

where we used the Pythagoras theorem(Lemma 2.11) and the fact that $\mathbb{E}\big[\mathbb{E}[f|\mathcal{B}_{i-1}]\big|\mathcal{B}_i\big] = \mathbb{E}[f|\mathcal{B}_{i-1}]$ since $\mathcal{B}_i \preceq \mathcal{B}_{i-1}$. So

$$\begin{aligned}\kappa^2 \leq |\mathbb{E}[g \cdot e(P_i)]|^2 &= \big|\mathbb{E}\big[\mathbb{E}[g \cdot e(P_i)|\mathcal{B}_i]\big]\big|^2 = \big|\mathbb{E}\big[e(P_i)\mathbb{E}[g|\mathcal{B}_i]\big]\big|^2 \\ &\leq \|\mathbb{E}[g|\mathcal{B}_i]\|_1^2 \leq \|\mathbb{E}[g|\mathcal{B}_i]\|_2^2 = \|\mathbb{E}[f|\mathcal{B}_i]\|_2^2 - \|\mathbb{E}[f|\mathcal{B}_{i-1}]\|_2^2.\end{aligned}$$

Thus the energy increases by κ^2 every step. But since the energy is bounded above by 1, the process should end in a finite number of steps $k \leq \frac{1}{\kappa^2}$. So $\|f - \mathbb{E}[f|\mathcal{B}_k]\|_{U^r} \leq \epsilon$, but since $\mathbb{E}[f|\mathcal{B}_k]$ is $\mathcal{B}_k$-measurable, we can write $\mathbb{E}[f|\mathcal{B}_k] = \tau(P_1, \cdots, P_k)$ for some function τ with $|\tau| = |\mathbb{E}[f|\mathcal{B}_k]| \leq |f| \leq 1$. ◀

We are now ready to prove Theorem 2.7.

Proof of Theorem 2.7. Recall that $\mathbb{K}$ is an extension field of dimension t over a prime field $\mathbb{F}_p$. The ϵ-net will be the set $\mathcal{N}$ of all functions of the form $\tau(P_1, \cdots, P_k)$ where $P_1, \cdots, P_k$ are degree $< r$ non-classical polynomials with zero constant terms, $\tau : \mathbb{T}^k \to \mathbb{C}$ is a bounded function and $k = k(\epsilon, p, r)$ is the constant given by Lemma 2.12. But we will not include all possible bounded $\tau : \mathbb{T}^k \to \mathbb{C}$. Firstly by Equation 1, $P_1, \cdots, P_k$ take values only in $\frac{1}{p^r}\mathbb{Z}/\mathbb{Z}$. Next we will discretize the set $\{z \in \mathbb{C} : |z| \leq 1\}$ into the ϵ-lattice i.e. we will only consider maps $\tau : (\frac{1}{p^r}\mathbb{Z}/\mathbb{Z})^k \to \{z \in \mathbb{C} : |z| \leq 1\} \cap \epsilon(\mathbb{Z} + i\mathbb{Z})$. The number of such maps is bounded by $(4/\epsilon^2)^{p^{rk}}$.

By Equation 1, a non-classical polynomial of degree $< r$ in n variables with zero constant term can be represented by $\leq \binom{nt+r-1}{r-1} r$ coefficients in $\{0, 1, \cdots, p-1\}$. So the number of such non-classical polynomials is bounded by $\exp\left(O_{r,\mathbb{K}}(n^{r-1})\right)$. Combining both the bounds,

$$|\mathcal{N}| \leq \exp\left(O_{r,\mathbb{K}}(n^{r-1})\right)^k \cdot (4/\epsilon^2)^{p^{rk}} = \exp\left(O_{\epsilon,\mathbb{K},r}(n^{r-1})\right).$$

We will now prove that $\mathcal{N}$ is a 3ϵ-net. Given any $f : \mathbb{K}^n \to [-1, 1]$, using Lemma 2.12, there is a function $\tau(P_1, \cdots, P_k)$ such that

$$\|f - \tau(P_1, P_2, \cdots, P_k)\|_{U^r} \leq \epsilon.$$

If we consider the $\tilde{\tau} \in \mathcal{N}$ by rounding values real and imaginary parts of τ to the nearest multiple of ϵ, we get

$$\begin{aligned}
&\|f - \tilde{\tau}(P_1, P_2, \cdots, P_k)\|_{U^r} \\
&\leq \|f - \tau(P_1, P_2, \cdots, P_k)\|_{U^r} + \|\tau(P_1, P_2, \cdots, P_k) - \tilde{\tau}(P_1, P_2, \cdots, P_k)\|_{U^r} \\
&\leq \epsilon + \|\tau(P_1, P_2, \cdots, P_k) - \tilde{\tau}(P_1, P_2, \cdots, P_k)\|_\infty \leq 3\epsilon.
\end{aligned}$$

◀

3 Locally Correctable Codes

We begin by defining locally correctable codes formally. Note that the definition below differs from the conventional one in terms of a local correction algorithm and adversarial errors (see, for instance, [39]); however, our definition is certainly weaker. Therefore, this makes our lower bounds stronger.

▶ **Definition 3.1** (Locally Correctable Code (LCC)). An (r, δ, τ) LCC is a code $\mathcal{C} \subset \Sigma^{\mathcal{X}}$ with the following property:
For each $x \in \mathcal{X}$ there is a distribution $\mathcal{M}_x$ over r-tuples of distinct[6] coordinates such that whenever $\tilde{f} \in \Sigma^{\mathcal{X}}$ is δ-close to some codeword $f \in \mathcal{C}$ in Hamming distance,

$$\Pr_{(y_1,\cdots,y_r)\sim\mathcal{M}_x}[\mathcal{D}_{x,y_1,\cdots,y_r}(\tilde{f}(y_1), \tilde{f}(y_2), \cdots, \tilde{f}(y_r)) = f(x)] \geq 1 - \tau$$

where $\mathcal{D}_{x,y_1,\cdots,y_r} : \Sigma^r \to \Sigma$, called the decoding operator, depends only on $x, y_1, \cdots, y_r$. If furthermore $\mathcal{X}$ is a vector space and $\mathcal{C}$ is affine invariant then we call it an affine invariant LCC.

[6] Without loss of generality, we can assume the tuples have distinct coordinates by adding dummy coordinates and modifying the decoding functions $\mathcal{D}_{x,y_1,\cdots,y_r}$

▶ Remark. Let $|\Sigma| = m$, Without loss of generality, we can assume that $\Sigma = \{1, 2, \cdots, m\}$. Then we can extend functions $f : \mathcal{X} \to \Sigma$ to $\hat{f} : \mathcal{X} \to \blacktriangle_m$. The decoding operators $\mathcal{D} : \Sigma^r \to \Sigma$ can also be extended to $\widehat{\mathcal{D}} : \blacktriangle_m^r \to \blacktriangle_m$ as follows: For $z_1, \cdots, z_r \in \blacktriangle_m$ define

$$\widehat{\mathcal{D}}(z_1, \cdots, z_r) = \sum_{1 \le i_1, \cdots, i_r \le m} e_{\mathcal{D}(i_1, \cdots, i_r)} (z_1)_{i_1} \cdots (z_r)_{i_r}$$

where e_j stands for the j^{th} coordinate vector in $\mathbb{R}^m$ and $(z_j)_i$ is the i^{th} coordinate of the vector z_j. Now we can rewrite the decoding condition as:

$$\mathbb{E}_{(y_1, \cdots, y_r) \sim \mathcal{M}_x} \left[\left\langle \hat{f}(x), \widehat{\mathcal{D}}_{x, y_1, \cdots, y_r} (\hat{f}(y_1), \hat{f}(y_2), \cdots, \hat{f}(y_r)) \right\rangle \right] \ge 1 - \tau.$$

First, we make the observation that any LCC must have good minimum distance.

▶ **Lemma 3.2.** *Let $\mathcal{C} \subset \Sigma^{\mathcal{X}}$ be an (r, δ, τ) LCC with $\tau < 1/2$, then the minimum distance of $\mathcal{C}$ is at least 2δ.*

Proof. Let $f, g \in \mathcal{C}$ be two distinct codewords such that $\Delta(f, g) < 2\delta$. Let h be the midpoint of f and g i.e. h is δ-close to both f and g. Let $x \in \mathcal{X}$ be such that $f(x) \neq g(x)$. By the LCC property,

$$\Pr_{(y_1, \cdots, y_r) \sim \mathcal{M}_x} [f(x) = \mathcal{D}_{x, y_1, \cdots, y_r} (h(y_1), \cdots, h(y_r))] \ge 1 - \tau$$

$$\Pr_{(y_1, \cdots, y_r) \sim \mathcal{M}_x} [g(x) = \mathcal{D}_{x, y_1, \cdots, y_r} (h(y_1), \cdots, h(y_r))] \ge 1 - \tau.$$

This is a contradiction when $\tau < \frac{1}{2}$. Therefore every two codewords must be at least 2δ apart. ◀

Now, we are ready to prove our main result of this section.

▶ **Theorem 3.3** (Lower bound for LCCs). *Let $\mathcal{C} \subset \Sigma^{\mathbb{K}^n}$ be an (r, δ, τ) affine-invariant LCC where $\tau < \frac{2\delta}{3}$. Then $|\mathcal{C}| \le \exp\left(O_{\delta, \mathbb{K}, r, |\Sigma|}(n^{r-1})\right)$.*

Proof. Let $|\Sigma| = m$. Let $\mathcal{N}$ be an $\epsilon/2$-net for the space of all bounded functions $\{f : \mathbb{K}^n \to \mathbb{C}\}$ with the metric induced by $\|\cdot\|_{U^r}$-norm where $\epsilon = \frac{2\delta}{3m^r}$. Given a bounded $f : \mathbb{K}^n \to \mathbb{C}$, define

$$\phi(f) := \text{argmin}_{h \in \mathcal{N}} \|f - h\|_{U^r}$$

(break ties arbitrarily). Since $\mathcal{N}$ is an $\epsilon/2$ net, we have $\|f - \phi(f)\|_{U^r} \le \epsilon/2$. Define $\Psi : \mathcal{C} \to \mathcal{N}^m$ as

$$\Psi(f) := (\phi(\hat{f}_1), \cdots, \phi(\hat{f}_m))$$

where $\hat{f}_i : \mathbb{K}^n \to \mathbb{R}_{\ge 0}$ is the i^{th} coordinate function of the simplex extension $\hat{f} : \mathbb{K}^n \to \blacktriangle_m$ of f. We claim that Ψ is one-one which implies that $|\mathcal{C}| \le |\mathcal{N}|^m$. Now using Theorem 2.7, the required bound follows. Suppose that Ψ is not one-one. Let $f, g \in \mathcal{C}$ be two distinct codewords such that $\Psi(f) = \Psi(g)$. This implies that

$$\forall\ i \in [m]\ \|\hat{f}_i - \hat{g}_i\|_{U^r} \le \|\hat{f}_i - \phi(\hat{f}_i)\|_{U^r} + \|\hat{g}_i - \phi(\hat{g}_i)\|_{U^r} \le \epsilon.$$

By affine invariance of $\mathcal{C}$, $f \circ \ell \in \mathcal{C}$ for all invertible affine maps $\ell : \mathbb{K}^n \to \mathbb{K}^n$. So by the local correction property,

$$\Pr_{\ell, y_0, (y_1, \cdots, y_r) \sim \mathcal{M}_{y_0}} [f \circ \ell(y_0) = \mathcal{D}_{y_0, y_1, \cdots, y_r} (f \circ \ell(y_1), \cdots, f \circ \ell(y_r))] \ge 1 - \tau$$

where ℓ ranges uniformly over all invertible affine maps from $\mathbb{K}^n \to \mathbb{K}^n$ and y_0 ranges uniformly over $\mathbb{K}^n$. Now consider the following difference:

$$\begin{aligned}
&\Pr_{\ell,y_0,(y_1,\cdots,y_r)\sim\mathcal{M}_{y_0}}[f\circ\ell(y_0) = \mathcal{D}_{y_0,y_1,\cdots,y_r}(f\circ\ell(y_1),\cdots,f\circ\ell(y_r))] \\
&\qquad - \Pr_{\ell,y_0,(y_1,\cdots,y_r)\sim\mathcal{M}_{y_0}}[g\circ\ell(y_0) = \mathcal{D}_{y_1,\cdots,y_r}(f\circ\ell(y_1),\cdots,f\circ\ell(y_r))] \\
&= \mathbb{E}_\ell\mathbb{E}_{y_0}\mathbb{E}_{(y_1,\cdots,y_r)\sim\mathcal{M}_{y_0}}\left[\left\langle \hat{f}\circ\ell(y_0), \widehat{\mathcal{D}}_{y_0,y_1,\cdots,y_r}(\hat{f}\circ\ell(y_1),\cdots,\hat{f}\circ\ell(y_r))\right\rangle\right. \\
&\qquad\qquad \left. - \left\langle \hat{g}\circ\ell(y_0), \widehat{\mathcal{D}}_{y_1,\cdots,y_r}(\hat{f}\circ\ell(y_1),\cdots,\hat{f}\circ\ell(y_r))\right\rangle\right] \\
&= \mathbb{E}_{y_0}\mathbb{E}_{(y_1,\cdots,y_r)\sim\mathcal{M}_{y_0}}\left[\mathbb{E}_\ell\left[\left\langle \hat{f}\circ\ell(y_0) - \hat{g}\circ\ell(y_0), \widehat{\mathcal{D}}_{y_0,y_1,\cdots,y_r}(\hat{f}\circ\ell(y_1),\cdots,\hat{f}\circ\ell(y_r))\right\rangle\right]\right]
\end{aligned}$$

Now we fix $y_0, y_1, \cdots, y_r$ and show that inner expectation is small for each tuple $(y_0, y_1, \cdots, y_r)$. Let us denote $\mathcal{D} = \mathcal{D}_{y_0,y_1,\cdots,y_r}$ for brevity. Let $t = \text{rank}(y_0, y_1, \cdots, y_r)$[7], thus there exist independent vectors $v_1, \cdots, v_t \in \mathbb{K}^n$ such that for every $0 \le i \le r$, $y_i = \sum_{j=1}^t \lambda_{ij} v_j$ for some fixed $\lambda_{ij} \in \mathbb{K}$. The action of a random invertible affine map ℓ can be approximated by sampling $z_0, z_1, \cdots, z_t \in \mathbb{K}^n$ uniformly and mapping $y_i \mapsto z_0 + \sum_{j=1}^t \lambda_{ij} z_j$ since with probability $1 - o_n(1)$, $z_1, \cdots, z_t$ will be independent. Therefore,

$$\begin{aligned}
&\mathbb{E}_\ell\left[\left\langle \hat{f}\circ\ell(y_0) - \hat{g}\circ\ell(y_0), \widehat{\mathcal{D}}_{y_0,y_1,\cdots,y_r}(\hat{f}\circ\ell(y_1),\cdots,\hat{f}\circ\ell(y_r))\right\rangle\right] \\
&= o_n(1) + \mathbb{E}_{z_0,z_1,\cdots,z_t\in\mathbb{K}^n}\left[\left\langle (\hat{f}-\hat{g})(z_0 + \sum_{j=1}^t \lambda_{0j} z_j),\right.\right. \\
&\qquad\qquad \left.\left. \widehat{\mathcal{D}}\left(\hat{f}(z_0 + \sum_{j=1}^t \lambda_{1j} z_j),\cdots,\hat{f}(z_0 + \sum_{j=1}^t \lambda_{rj} z_j)\right)\right\rangle\right] \\
&\qquad\qquad\qquad\qquad \text{(we can ignore the } o_n(1) \text{ term)} \\
&= \mathbb{E}_{z_0,z_1,\cdots,z_t\in\mathbb{K}^n}\left[\left\langle (\hat{f}-\hat{g})(z_0 + \sum_{j=1}^t \lambda_{0j} z_j),\right.\right. \\
&\qquad\qquad \left.\left. \left(\sum_{1\le i_1,\cdots,i_r\le m} e_{\mathcal{D}(i_1,\cdots,i_r)} \prod_{k=1}^r \hat{f}_{i_k}(z_0 + \sum_{j=1}^t \lambda_{kj} z_j)\right)\right\rangle\right] \\
&= \mathbb{E}_{z_0,z_1,\cdots,z_t\in\mathbb{K}^n}\left[\left(\sum_{1\le i_1,\cdots,i_r\le m}\right.\right. \\
&\qquad\qquad \left.\left. (\hat{f}-\hat{g})_{\mathcal{D}(i_1,\cdots,i_r)}(z_0 + \sum_{j=1}^t \lambda_{0j} z_j)\cdot \prod_{k=1}^r \hat{f}_{i_k}(z_0 + \sum_{j=1}^t \lambda_{kj} z_j)\right)\right] \\
&\le \left(\sum_{0\le i_1,\cdots,i_r\le m-1} \|(\hat{f}-\hat{g})_{\mathcal{D}(i_1,\cdots,i_r)}\|_{U^r}\right) \le m^r\epsilon
\end{aligned}$$

where the first inequality is obtained by applying generalized von Neumann inequality

[7] $\text{rank}(y_0, y_1, \cdots, y_r)$ is the dimension of the subspace spanned by the vectors $y_0, y_1, \cdots, y_r$.

(Lemma 2.6) to each term. Therefore

$$\begin{aligned}
&\Pr_{\ell,y_0,(y_1,\cdots,y_r)\sim\mathcal{M}_{y_0}}[g\circ\ell(y_0)=\mathcal{D}_{y_1,\cdots,y_r}(f\circ\ell(y_1),\cdots,f\circ\ell(y_r))]\\
&\geq \Pr_{\ell,y_0,(y_1,\cdots,y_r)\sim\mathcal{M}_{y_0}}[f\circ\ell(y_0)=\mathcal{D}_{y_1,\cdots,y_r}(f\circ\ell(y_1),\cdots,f\circ\ell(y_r))]-m^r\epsilon\\
&\geq 1-\tau-2\delta/3.
\end{aligned}$$

On the other hand,

$$\begin{aligned}
&\Pr_{\ell,y_0,(y_1,\cdots,y_r)\sim\mathcal{M}_{y_0}}[g\circ\ell(y_0)=\mathcal{D}_{y_1,\cdots,y_r}(f\circ\ell(y_1),\cdots,f\circ\ell(y_r))]\\
&\leq \Pr_{\ell,y_0,(y_1,\cdots,y_r)\sim\mathcal{M}_{y_0}}[g\circ\ell(y_0)=f\circ\ell(y_0)]\\
&\qquad\qquad+\Pr_{\ell,y_0,(y_1,\cdots,y_r)\sim\mathcal{M}_{y_0}}[f\circ\ell(y_0)\neq\mathcal{D}_{y_1,\cdots,y_r}(f\circ\ell(y_1),\cdots,f\circ\ell(y_r))]\\
&\leq \Pr_x[f(x)=g(x)]+\tau\leq 1-2\delta+\tau \qquad \text{(By Lemma 3.2)}
\end{aligned}$$

This is a contradiction when $\tau<\frac{2\delta}{3}$. ◀

4 Locally Testable Codes

We start by defining locally testable codes in a formulation convenient for our use.

▶ **Definition 4.1** (Locally Testable Code (LTC)). An (r,δ,τ) LTC is a code $\mathcal{C}\subset\Sigma^{\mathcal{X}}$ with minimum distance at least δ and the following property:
There is a distribution $\mathcal{M}$ over r-tuples of distinct[8] coordinates such that for each codeword $f\in C$,

$$\Pr_{(y_1,\cdots,y_r)\sim\mathcal{M}}[\mathcal{D}_{y_1,\cdots,y_r}(f(y_1),f(y_2),\cdots,f(y_r))=1]\geq 3/4$$

and for every $g\in\Sigma^{\mathcal{X}}$ which is τ-far away from every codeword,

$$\Pr_{(y_1,\cdots,y_r)\sim\mathcal{M}}[\mathcal{D}_{y_1,\cdots,y_r}(g(y_1),g(y_2),\cdots,g(y_r))=1]\leq 1/4$$

where $\mathcal{D}_{y_1,\cdots,y_r}:\Sigma^r\to\{0,1\}$, called the testing operator, depends only on $y_1,\cdots,y_r$. If furthermore $\mathcal{X}$ is a vector space and $\mathcal{C}$ is affine-invariant then we call it an affine invariant LTC.

▶ Remark. Let $|\Sigma|=m$, Without loss of generality, we can assume that $\Sigma=\{1,2,\cdots,m\}$. We can extend $f:\mathcal{X}\to\Sigma$ to $\hat{f}:\mathcal{X}\to\blacktriangle_m$. The testing operator $\mathcal{D}:\Sigma^r\to\{0,1\}$ can also be extended to $\widehat{\mathcal{D}}:\blacktriangle_m^r\to[0,1]$ as follows: For $z_1,\cdots,z_r\in\blacktriangle_m$ define

$$\widehat{\mathcal{D}}(z_1,\cdots,z_r)=\sum_{1\leq i_1,\cdots,i_r\leq m}\mathcal{D}(i_1,\cdots,i_r)(z_1)_{i_1}\cdots(z_r)_{i_r}. \tag{2}$$

Now we can rewrite the probability in terms of expectation as:

$$\begin{aligned}
&\Pr_{(y_1,\cdots,y_r)\sim\mathcal{M}}[\mathcal{D}_{y_1,\cdots,y_r}(f(y_1),\cdots,f(y_r))=1]\\
&=\mathbb{E}_{(y_1,\cdots,y_r)\sim\mathcal{M}}[\widehat{\mathcal{D}}_{y_1,\cdots,y_r}(\hat{f}\circ\ell(y_1),\cdots,\hat{f}\circ\ell(y_r))].
\end{aligned}$$

[8] Again, without loss of generality, we can assume the tuples have distinct coordinates by adding dummy coordinates and modifying the decoding functions $\mathcal{D}_{y_1,\cdots,y_r}$

We are now ready to prove the main result of this section.

▶ **Theorem 4.2** (Lower bound for LTC's). *Let $\mathcal{C} \subset \Sigma^{\mathbb{K}^n}$ be an $(r, \delta, \delta/3)$ affine invariant LTC, then $|\mathcal{C}| \leq \exp\left(O_{\delta,\mathbb{K},r,|\Sigma|}(n^{r-2})\right)$.*

Proof. Let $|\Sigma| = m$. The proof is very similar to that of Theorem 3.3. Let $\mathcal{N}$ be an $\epsilon/2$-net for the space of all bounded functions $\{f : \mathbb{K}^n \to \mathbb{C}\}$ with the metric induced by $\|\cdot\|_{U^{r-1}}$-norm where $\epsilon = 1/2rm^r$. Define $\Psi : \mathcal{C} \to \mathcal{N}^m$ as in the proof of Theorem 3.3, it is enough to show that Ψ is one-one. Suppose that Ψ is not one-one. Then there exists $f, g \in \mathcal{C}$ which are distinct such that $\Psi(f) = \Psi(g)$. This implies that

$$\forall\ i \in [m]\ \|\hat{f}_i - \hat{g}_i\|_{U^{r-1}} \leq \epsilon.$$

By affine invariance of $\mathcal{C}$, $f \circ \ell \in \mathcal{C}$ for all invertible affine maps $\ell : \mathbb{K}^n \to \mathbb{K}^n$. So

$$\mathbb{E}_\ell \mathbb{E}_{(y_1,\cdots,y_r)\sim\mathcal{M}}[\mathcal{D}_{y_1,\cdots,y_r}(f \circ \ell(y_1), f \circ \ell(y_2), \cdots, f \circ \ell(y_r))] \geq 3/4$$

where ℓ ranges over all invertible affine maps from $\mathbb{K}^n \to \mathbb{K}^n$. Let $H \in \Sigma^{\mathcal{X}}$ be a random word where for each coordinate $x \in \mathcal{X}$ independently,

$$H(x) = \begin{cases} f(x) & \text{with probability } 1/2 \\ g(x) & \text{with probability } 1/2 \end{cases}.$$

Define $\hat{h} : \mathcal{X} \to \blacktriangle_m$ as $\hat{h}(x) = \mathbb{E}_H[\widehat{H}(x)] = \frac{\hat{f}(x)+\hat{g}(x)}{2}$ where $\hat{f}, \hat{g}$ are the simplex extensions of the original f, g. So $\forall\ i \in [m]\ \|\hat{f}_i - \hat{h}_i\|_{U^{r-1}} = \|\hat{f}_i - \hat{g}_i\|_{U^{r-1}}/2 \leq \epsilon/2$. We will now show that the test accepts $H \circ \ell$ with good probability when ℓ is a random invertible affine map from $\mathbb{K}^n \to \mathbb{K}^n$.

$$\begin{aligned}
&\mathbb{E}_H \mathbb{E}_\ell \mathbb{E}_{(y_1,\cdots,y_r)\sim\mathcal{M}}[\mathcal{D}_{y_1,\cdots,y_r}(f \circ \ell(y_1), \cdots, f \circ \ell(y_r)) \\
&\qquad\qquad - \mathcal{D}_{y_1,\cdots,y_r}(H \circ \ell(y_1), \cdots, H \circ \ell(y_r))] \\
&= \mathbb{E}_H \mathbb{E}_\ell \mathbb{E}_{(y_1,\cdots,y_r)\sim\mathcal{M}}[\widehat{\mathcal{D}}_{y_1,\cdots,y_r}(\hat{f} \circ \ell(y_1), \cdots, \hat{f} \circ \ell(y_r)) \\
&\qquad\qquad - \widehat{\mathcal{D}}_{y_1,\cdots,y_r}(\widehat{H} \circ \ell(y_1), \cdots, \widehat{H} \circ \ell(y_r))] \\
&= \mathbb{E}_\ell \mathbb{E}_{(y_1,\cdots,y_r)\sim\mathcal{M}}[\widehat{\mathcal{D}}_{y_1,\cdots,y_r}(\hat{f} \circ \ell(y_1), \cdots, \hat{f} \circ \ell(y_r)) \\
&\qquad\qquad - \widehat{\mathcal{D}}_{y_1,\cdots,y_r}(\hat{h} \circ \ell(y_1), \cdots, \hat{h} \circ \ell(y_r))] \\
&\quad \text{(using multilinear expansion of } \widehat{\mathcal{D}}_{y_1,\cdots,y_r} \text{(Equation 2) and taking expectation over } H) \\
&= \mathbb{E}_{(y_1,\cdots,y_r)\sim\mathcal{M}}\left[\mathbb{E}_\ell\left[\widehat{\mathcal{D}}_{y_1,\cdots,y_r}(\hat{f} \circ \ell(y_1), \cdots, \hat{f} \circ \ell(y_r)) \right.\right. \\
&\qquad\qquad \left.\left. -\widehat{\mathcal{D}}_{y_1,\cdots,y_r}(\hat{h} \circ \ell(y_1), \cdots, \hat{h} \circ \ell(y_r))\right]\right]
\end{aligned}$$

Now we fix $y_1, \cdots, y_r$ and show that inner expectation is small for each tuple $(y_1, \cdots, y_r)$. Let us denote $\mathcal{D} = \mathcal{D}_{y_1,\cdots,y_r}$ for brevity. Let $t = \text{rank}(y_1, \cdots, y_r)$, thus there exist independent vectors $v_1, \cdots, v_t \in \mathbb{K}^n$ such that for every $1 \leq i \leq r$, $y_i = \sum_{j=1}^t \lambda_{ij} v_j$ for some fixed $\lambda_{ij} \in \mathbb{K}$. The action of a random invertible affine map ℓ can be approximated by sampling $z_0, z_1, \cdots, z_t \in \mathbb{K}^n$ uniformly and mapping $y_i \mapsto z_0 + \sum_{j=1}^t \lambda_{ij} z_j$ since with probability

$1 - o_n(1)$, $z_1, \cdots, z_t$ will be independent. Therefore,

$$\begin{aligned}
&\mathbb{E}_\ell \left[\widehat{\mathcal{D}}_{y_1,\cdots,y_r}(\hat{f} \circ \ell(y_1), \cdots, \hat{f} \circ \ell(y_r)) - \widehat{\mathcal{D}}_{y_1,\cdots,y_r}(\hat{h} \circ \ell(y_1), \cdots, \hat{h} \circ \ell(y_r)) \right] \\
&= o_n(1) + \mathbb{E}_{z_0,\cdots,z_t \in \mathbb{K}^n} \left[\widehat{\mathcal{D}}(\hat{f}(z_0 + \sum_{j=1}^{t} \lambda_{1j} z_j), \cdots, \hat{f}(z_0 + \sum_{j=1}^{t} \lambda_{rj} z_j)) \right. \\
&\qquad \left. - \mathcal{D}(\hat{h}(z_0 + \sum_{j=1}^{t} \lambda_{1j} z_j), \cdots, \hat{h}(z_0 + \sum_{j=1}^{t} \lambda_{rj} z_j)) \right] \\
&= \mathbb{E}_{z_0,z_1,\cdots,z_t \in \mathbb{K}^n} \left[\sum_{1 \le i_1,\cdots,i_r \le m} \right. \\
&\qquad \left. \mathcal{D}(i_1, \cdots, i_r) \left(\prod_{k=1}^{r} \hat{f}_{i_k}(z_0 + \sum_{j=1}^{t} \lambda_{kj} z_j) - \prod_{k=1}^{r} \hat{h}_{i_k}(z_0 + \sum_{j=1}^{t} \lambda_{kj} z_j) \right) \right] \\
&\le r \cdot m^r \cdot \frac{\epsilon}{2} = \frac{1}{4}
\end{aligned}$$

where the last line is obtained by forming hybrids i.e. writing

$$\begin{aligned}
&\hat{f}_{i_1} \cdot \hat{f}_{i_2} \cdots \hat{f}_{i_r} - \hat{h}_{i_1} \cdot \hat{h}_{i_2} \cdots \hat{h}_{i_r} \\
&= (\hat{f}_{i_1} - \hat{h}_{i_1}) \cdot \hat{f}_{i_2} \cdots \hat{f}_{i_r} + \hat{h}_{i_1} \cdot (\hat{f}_{i_2} - \hat{h}_{i_2}) \cdots \hat{f}_{i_r} + \cdots + \hat{h}_{i_1} \cdot \hat{h}_{i_2} \cdots (\hat{f}_{i_r} - \hat{h}_{i_r})
\end{aligned}$$

and using Lemma 2.6 for each term. Therefore

$$\begin{aligned}
&\mathbb{E}_H \mathbb{E}_\ell \mathbb{E}_{(y_1,\cdots,y_r) \sim \mathcal{M}} [\mathcal{D}_{y_1,\cdots,y_r}(H \circ \ell(y_1), \cdots, H \circ \ell(y_r))] \\
&\ge \mathbb{E}_\ell \mathbb{E}_{(y_1,\cdots,y_r) \sim \mathcal{M}} [\mathcal{D}_{y_1,\cdots,y_r}(f \circ \ell(y_1), \cdots, f \circ \ell(y_r))] - \frac{1}{4} \ge \frac{3}{4} - \frac{1}{4} = \frac{1}{2}.
\end{aligned}$$

By Markov inequality,

$$\begin{aligned}
\frac{1}{4} &\le \Pr_H \left[\mathbb{E}_\ell \mathbb{E}_{(y_1,\cdots,y_r) \sim \mathcal{M}} [\mathcal{D}_{y_1,\cdots,y_r}(H \circ \ell(y_1), \cdots, H \circ \ell(y_r))] \ge \frac{1}{3} \right] \\
&\le \Pr_H \left[\exists \ell \;\; \mathbb{E}_{(y_1,\cdots,y_r) \sim \mathcal{M}} [\mathcal{D}_{y_1,\cdots,y_r}(H \circ \ell(y_1), \cdots, H \circ \ell(y_r))] \ge \frac{1}{3} \right] \\
&\le \Pr_H \left[\exists \ell \;\; \Delta(H \circ \ell, \mathcal{C})] \le \frac{\delta}{3} \right] && \text{(by the soundness of the tester)} \\
&= \Pr_H \left[\Delta(H, \mathcal{C})] \le \frac{\delta}{3} \right] && \text{(since } \ell \text{ is invertible and } \mathcal{C} \text{ is affine invariant)}
\end{aligned}$$

Let $\mathcal{H} = \mathrm{Supp}(H)$ be the set of words between f and g i.e. the set of all words $e \in \Sigma^{\mathbb{K}^n}$ such that $e(x) = f(x)$ or $e(x) = g(x)$ for all $x \in \mathbb{K}^n$. We have $|\mathcal{H}| = 2^{\Delta(f,g)n}$. Since the distribution of H is uniform in $\mathcal{H}$, we proved that at least $\frac{1}{4}$ fraction of words in $\mathcal{H}$ contain a codeword in their $\delta/3$ neighborhood, let $\mathcal{H}' \subset \mathcal{H}$ denote this subset. Therefore the $\delta/6$ neighborhoods around the points in $\mathcal{H}'$ must be disjoint or else two distinct codewords will be $< \delta$ close to each other. The number of words in $\mathcal{H}$ which lie in a Hamming ball of radius $\delta/6$ around a point of $\mathcal{H}'$ is

$$\sum_{i=0}^{\delta n/6} \binom{\Delta(f,g)n}{i} \ge 2^{H(\delta/6\Delta(f,g))\Delta(f,g)n - o(n)} \ge 2^{H(\delta/6)\Delta(f,g)n - o(n)}$$

where $H(\cdot)$ is the binary entropy function. By a packing argument, we can upper bound the size of $\mathcal{H}'$ as

$$|\mathcal{H}'| \leq \frac{2^{\Delta(f,g)n}}{2^{H(\delta/6)\Delta(f,g)n-o(n)}} = o(|\mathcal{H}|).$$

This contradicts the fact that $|\mathcal{H}'| \geq |\mathcal{H}|/4$. ◀

5 Concluding Remarks

In this work, we proved tight lower bounds for constant query affine-invariant LCCs and LTCs when the number of queries r, underlying field $\mathbb{K}$ and the alphabet Σ are constant. However the constants in the bounds we obtain are of Ackermann-type in $r, |\mathbb{K}|, |\Sigma|$ because of the use of higher-order Fourier analysis. Improving the dependence on these parameters is an open problem which might require new ideas. In a recent work, Bhowmick and Lovett [11] obtain a "bias implies low rank" theorem for polynomials over growing fields. This might be a first step towards proving a variant of the inverse Gowers theorem (Lemma 2.5) for growing field size, which could then be used to make our lower bounds extend to the case of growing field size.

We also remark that our lower bounds work for any LCC or LTC where the queries are obtained as fixed linear combinations of uniformly chosen points from $\mathbb{K}^n$. Affine-invariant codes are a natural class of local codes where this is true. Relaxing these conditions to get lower bounds for a more general class of LCCs or LTCs is an open problem.

Acknowledgements. We thank Madhu Sudan for helpful pointers to previous work. The second author would like to thank his advisor, Zeev Dvir, for his guidance and encouragement.

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, 1998.

2 Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: A new characterization of NP. *J. ACM*, 45(1):70–122, 1998.

3 Boaz Barak, Zeev Dvir, Amir Yehudayoff, and Avi Wigderson. Rank bounds for design matrices with applications to combinatorial geometry and locally correctable codes. In *Proc. 43rd Annual ACM Symposium on the Theory of Computing*, pages 519–528. ACM, 2011.

4 Omer Barkol, Yuval Ishai, and Enav Weinreb. On locally decodable codes, self-correctable codes, and t-private PIR. In *Proc. 11th International Workshop on Randomization and Computation*, pages 311–325. Springer, 2007.

5 Eli Ben-Sasson, Noga Ron-Zewi, and Madhu Sudan. Sparse affine-invariant linear codes are locally testable. In *Proc. 53rd Annual IEEE Symposium on Foundations of Computer Science*, pages 561–570. IEEE, 2012.

6 Eli Ben-Sasson and Madhu Sudan. Robust locally testable codes and products of codes. In *Proc. 8th International Workshop on Randomization and Computation*, pages 286–297, 2004.

7 Eli Ben-Sasson and Madhu Sudan. Short PCPs with polylog query complexity. *SIAM J. on Comput.*, 38(2):551–607, 2008.

8 Eli Ben-Sasson and Madhu Sudan. Limits on the rate of locally testable affine-invariant codes. In *Proc. 15th International Workshop on Randomization and Computation*, pages 412–423. Springer, 2011.

9 Arnab Bhattacharyya and Abhishek Bhowmick. Using higher-order Fourier analysis over general fields. *Preprint arXiv:1505.00619*, 2015.

10 Arnab Bhattacharyya, Zeev Dvir, Amir Shpilka, and Shubhangi Saraf. Tight lower bounds for 2-query LCCs over finite fields. In *Proc. 52nd Annual IEEE Symposium on Foundations of Computer Science*, pages 638–647. IEEE, 2011.

11 Abhishek Bhowmick and Shachar Lovett. Bias vs structure of polynomials in large fields, and applications in effective algebraic geometry and coding theory. *Preprint arXiv:1506.02047*, 2015.

12 Abhishek Bhowmick and Shachar Lovett. The list decoding radius of Reed-Muller codes over small fields. In *Proc. 47th Annual ACM Symposium on the Theory of Computing*, pages 277–285, 2015. doi:10.1145/2746539.2746543.

13 Manuel Blum and Sampath Kannan. Designing programs that check their work. *J. ACM*, 42(1):269–291, 1995.

14 Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-testing/correcting with applications to numerical problems. *J. Comp. Sys. Sci.*, 47(3):549–595, 1993.

15 Benny Chor, Eyal Kushilevitz, Oded Goldreich, and Madhu Sudan. Private information retrieval. *J. ACM*, 45(6):965–981, 1998.

16 Irit Dinur. The PCP theorem by gap amplification. *J. ACM*, 54(3):12, 2007.

17 Zeev Dvir, Shubhangi Saraf, and Avi Wigderson. Breaking the quadratic barrier for 3-LCC's over the reals. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, pages 784–793. ACM, 2014.

18 Zeev Dvir and Amir Shpilka. Locally decodable codes with two queries and polynomial identity testing for depth 3 circuits. *SIAM J. on Comput.*, 36(5):1404–1434, 2007.

19 Oded Goldreich, Howard Karloff, Leonard J Schulman, and Luca Trevisan. Lower bounds for linear locally decodable codes and private information retrieval. In *Proc. 17th Annual IEEE Conference on Computational Complexity*, pages 143–151. IEEE, 2002.

20 Oded Goldreich and Madhu Sudan. Locally testable codes and PCPs of almost-linear length. *J. ACM*, 53(4):558–655, July 2006.

21 William T Gowers. A new proof of Szemerédi's theorem. *Geom. Funct. Anal.*, 11(3):465–588, 2001.

22 Ben Green. Montreal lecture notes on quadratic Fourier analysis. *Preprint arXiv:math/0604089*, 2006.

23 Alan Guo, Swastik Kopparty, and Madhu Sudan. New affine-invariant codes from lifting. In *Proc. 4th Innovations in Theoretical Computer Science*, pages 529–540. ACM, 2013.

24 Alan Xinyu Guo. Some closure features of locally testable affine-invariant properties. Master's thesis, Massachusetts Institute of Technology, 2013.

25 Venkatesan Guruswami, Madhu Sudan, Ameya Velingker, and Carol Wang. Limitations on testable affine-invariant codes in the high-rate regime. In *Proc. 26th ACM-SIAM Symposium on Discrete Algorithms*, pages 1312–1325. SIAM, 2015.

26 T. Kasami, S. Lin, and W.W. Peterson. Some results on cyclic codes which are invariant under the affine group and their applications. *Inform. and Comput.*, 11(5–6):475–496, 1967.

27 Jonathan Katz and Luca Trevisan. On the efficiency of local decoding procedures for error-correcting codes. In *Proc. 32nd Annual ACM Symposium on the Theory of Computing*, pages 80–86. ACM, 2000.

28 Tali Kaufman and Madhu Sudan. Algebraic property testing: the role of invariance. In *Proc. 40th Annual ACM Symposium on the Theory of Computing*, pages 403–412. ACM, 2008.

29 Iordanis Kerenidis and Ronald de Wolf. Exponential lower bound for 2-query locally decodable codes via a quantum argument. In *Proc. 35th Annual ACM Symposium on the Theory of Computing*, pages 106–115. ACM, 2003.

30 Richard J Lipton. Efficient checking of computations. In *Proc. 7th Symposium on Theoretical Aspects of Computer Science*, pages 207–215. Springer, 1990.

31 Or Meir. Combinatorial construction of locally testable codes. *SIAM J. on Comput.*, 39(2):491–544, 2009.

32 Madhu Sudan, Luca Trevisan, and Salil Vadhan. Pseudorandom generators without the XOR lemma. In *Proc. 31st Annual ACM Symposium on the Theory of Computing*, pages 537–546. ACM, 1999.

33 Terence Tao. *Higher order Fourier analysis*, volume 142. American Mathematical Soc., 2012.

34 Terence Tao and Tamar Ziegler. The inverse conjecture for the Gowers norm over finite fields in low characteristic. *Ann. Comb.*, 16(1):121–188, 2012.

35 Madhur Tulsiani and Julia Wolf. Quadratic Goldreich-Levin theorems. *SIAM Journal on Computing*, 43(2):730–766, 2014.

36 Michael Viderman. Explicit strong LTCs with inverse poly-log rate and constant soundness. *Electronic Colloquium on Computational Complexity (ECCC)*, 22:20, 2015. URL: `http://eccc.hpi-web.de/report/2015/020`.

37 David Woodruff. New lower bounds for general locally decodable codes. *Electronic Colloquium on Computational Complexity (ECCC)*, 14(006), 2007.

38 David P Woodruff. A quadratic lower bound for three-query linear locally decodable codes over any field. *J. Comput. Sci. Technol.*, 27(4):678–686, 2012.

39 Sergey Yekhanin. Locally decodable codes. In *Computer Science–Theory and Applications*, pages 289–290. Springer, 2011.

A Proof of generalized von Neumann inequality (Lemma 2.6)

Since the lemma is not stated in the form we want in [33], we will include a proof here for completeness. To prove Lemma 2.6, we need the following lemma first.

▶ **Lemma A.1** (Exercise 1.3.22 in [33]). *Let $f : \mathbb{K}^n \to \mathbb{C}$ be a function, and for each $1 \le i \le k$, let $g_i : (\mathbb{K}^n)^k \to \mathbb{C}$ be a bounded function which is independent of the i^{th} coordinate of $(\mathbb{K}^n)^k$. Then,*

$$|\mathbb{E}_{x_1,\cdots,x_k \in \mathbb{K}^n}[f(x_1+x_2+\cdots+x_k)\prod_{i=1}^{k} g_i(x_1,\cdots,x_k)]| \le \|f\|_{U^k}\,.$$

Proof. The proof is by induction on k and using Cauchy-Schwarz inequality repeatedly. The case $k=1$ is true by definition of $\|\cdot\|_{U^1}$.

$$\begin{aligned}
&\left|\mathbb{E}_{x_1,\cdots,x_k \in \mathbb{K}^n}\left[f(x_1+x_2+\cdots+x_k)\prod_{i=1}^{k} g_i(x_1,\cdots,x_k)\right]\right| \\
&= \left|\mathbb{E}_{x_2,\cdots,x_k}\left[g_1(x_1,\cdots,x_k)\mathbb{E}_{x_1}\left[f(x_1+x_2+\cdots+x_k)\prod_{i=2}^{k} g_i(x_1,\cdots,x_k)\right]\right]\right| \\
&\qquad\qquad \text{(since } g_1 \text{ doesn't depend on } x_1\text{)} \\
&\le \left|\mathbb{E}_{x_2,\cdots,x_k}\left[\mathbb{E}_{x_1'}\left[f(x_1'+x_2+\cdots+x_k)\prod_{i=2}^{k} g_i(x_1',x_2,\cdots,x_k)\right]\right.\right. \\
&\qquad\qquad \left.\left.\cdot\mathbb{E}_{x_1}\left[\bar{f}(x_1+x_2+\cdots+x_k)\prod_{i=2}^{k} \bar{g}_i(x_1,x_2,\cdots,x_k)\right]\right]\right|^{1/2} \\
&\qquad\qquad \text{(By Cauchy-Schwarz inequality and the fact that } |g_1| \le 1\text{)}
\end{aligned}$$

$$= |\mathbb{E}_{x_1,h_1}[\mathbb{E}_{x_2,\cdots,x_k}[\Delta_{h_1} f(x_1+x_2+\cdots+x_k) \cdot \prod_{i=2}^{k} g_i(x_1+h_1,x_2,\cdots,x_k)\bar{g}_i(x_1,x_2,\cdots,x_k)]]|^{1/2}$$

(By substituting $x_1' = x_1 + h_1$)

$$\leq \left|\mathbb{E}_{x_1,h_1}\left[\mathbb{E}_{h_2,\cdots,h_k,z}[\Delta_{h_k}\cdots\Delta_{h_1} f(x_1+z)]^{1/2^{k-1}}\right]\right|^{1/2}$$

(By induction hypothesis and the definition of Gowers norm)

$$\leq |\mathbb{E}_{x_1,h_1,h_2,\cdots,h_k,z}[\Delta_{h_k}\cdots\Delta_{h_1} f(x_1+z)]|^{1/2^k}$$ (By Jensen's inequality)

$$= |\mathbb{E}_{h_1,h_2,\cdots,h_k,z}[\Delta_{h_k}\cdots\Delta_{h_1} f(z)]|^{1/2^k} = \|f\|_{U^k}$$

◀

Proof of Lemma 2.6. By symmetry, it is enough to show that

$$|\mathbb{E}_{z_1,\cdots,z_m\in\mathbb{K}^n}[f_0(\mathcal{L}_0(z_1,\cdots,z_m))\prod_{i=1}^{k} f_i(\mathcal{L}_i(z_1,\cdots,z_m))]| \leq \|f_0\|_{U^k}.$$

We will make a linear change of variables so that we can use Lemma A.1 to get the required bound. For each $1 \leq i \leq k$, since $\mathcal{L}_0$ is not a multiple of $\mathcal{L}_i$, there exists a vector $v_i \in \mathbb{K}^m$ such that $\mathcal{L}_0(v_i) = 1$ and $\mathcal{L}_i(v_i) = 0$. Now we make the following change of variables: $(z_1,\cdots,z_m) \to (x_1,\cdots,x_m) + \sum_{i=1}^{k} y_i v_i^T$ where $x_1,\cdots,x_m$ and $y_1,\cdots,y_k$ are the new variables which range over $\mathbb{K}^n$.

$$|\mathbb{E}_{z_1,\cdots,z_m\in\mathbb{K}^n}[f_0(\mathcal{L}_0(z_1,\cdots,z_m))\prod_{i=1}^{k} f_i(\mathcal{L}_i(z_1,\cdots,z_m))]|$$

$$= \left|\mathbb{E}_{x_1,\cdots,x_m,y_1,\cdots,y_k\in\mathbb{K}^n}\left[f_0\left(\mathcal{L}_0(x_1,\cdots,x_m)+\sum_{j\in[k]} y_j\right)\prod_{i\in[k]} f_i\left(\mathcal{L}_i(x_1,\cdots,x_m)+\sum_{j\in[k]\setminus\{i\}} y_j\mathcal{L}_i(v_j)\right)\right]\right|$$

(By change of variables and linearity of $\mathcal{L}_i$)

$$\leq \mathbb{E}_{x_1,\cdots,x_m\in\mathbb{K}^n}\left[\left|\mathbb{E}_{y_1,\cdots,y_k\in\mathbb{K}^n}\left[f_0\left(\mathcal{L}_0(x_1,\cdots,x_m)+\sum_{j\in[k]} y_j\right)\prod_{i\in[k]} f_i\left(\mathcal{L}_i(x_1,\cdots,x_m)+\sum_{j\in[k]\setminus\{i\}} y_j\mathcal{L}_i(v_j)\right)\right]\right|\right]$$

$$\leq \|f_0\|_{U^k}$$ (By Lemma A.1)

◀

Degree and Sensitivity: Tails of Two Distributions

Parikshit Gopalan[1], Rocco A. Servedio[*2], and Avi Wigderson[†3]

1 **Microsoft Research, Redmond, USA**
parik@microsoft.com
2 **Department of Computer Science, Columbia University, New York, USA**
rocco@cs.columbia.edu
3 **Institute for Advanced Study, Princeton University, Princeton, USA**
avi@ias.edu

Abstract

The *sensitivity* of a Boolean function f is the maximum, over all inputs x, of the number of sensitive coordinates of x (namely the number of Hamming neighbors of x with different f-value). The well-known *sensitivity conjecture* of Nisan (see also Nisan and Szegedy) states that every sensitivity-s Boolean function can be computed by a polynomial over the reals of degree $\mathrm{poly}(s)$. The best known upper bounds on degree, however, are exponential rather than polynomial in s.

Our main result is an approximate version of the conjecture: every Boolean function with sensitivity s can be ϵ-approximated (in ℓ_2) by a polynomial whose degree is $s \cdot \mathrm{polylog}(1/\epsilon)$. This is the first improvement on the folklore bound of s/ϵ. We prove this via a new "switching lemma for low-sensitivity functions" which establishes that a random restriction of a low-sensitivity function is very likely to have low decision tree depth. This is analogous to the well-known switching lemma for AC^0 circuits.

Our proof analyzes the combinatorial structure of the graph G_f of sensitive edges of a Boolean function f. Understanding the structure of this graph is of independent interest as a means of understanding Boolean functions. We propose several new complexity measures for Boolean functions based on this graph, including *tree sensitivity* and *component dimension*, which may be viewed as relaxations of worst-case sensitivity, and we introduce some new techniques, such as *proper walks* and *shifting*, to analyze these measures. We use these notions to show that the graph of a function of full degree must be sufficiently complex, and that random restrictions of low-sensitivity functions are unlikely to lead to such complex graphs.

We postulate a robust analogue of the sensitivity conjecture: if **most** inputs to a Boolean function f have low sensitivity, then **most** of the Fourier mass of f is concentrated on small subsets. We prove a lower bound on tree sensitivity in terms of decision tree depth, and show that a polynomial strengthening of this lower bound implies the robust conjecture. We feel that studying the graph G_f is interesting in its own right, and we hope that some of the notions and techniques we introduce in this work will be of use in its further study.

1998 ACM Subject Classification F.1.3 Complexity Measures and Classes

Keywords and phrases Boolean functions, sensitivity, decision trees, Fourier degree

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.13

1 Introduction

The smoothness of a continuous function captures how gradually it changes locally (according to the metric of the underlying space). For Boolean functions on $\{0,1\}^n$, a natural analog is

* Partially supported by NSF grants CCF-1319788 and CCF-1420349.
† Partially supported by NSF grant CCF-1412958.

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 13; pp. 13:1–13:23

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

sensitivity, capturing how many neighbors of a point have different function values. More formally, the *sensitivity* of $f : \{0,1\}^n \to \{\pm 1\}$ at input $x \in \{0,1\}^n$, written $s(f,x)$, is the number of neighbors y of x in the Hamming cube $\{0,1\}^n$ such that $f(y) \neq f(x)$. The *max sensitivity* of f, written $s(f)$ and often referred to simply as the "sensitivity of f", is defined as $s(f) = \max_{x \in \{0,1\}^n} s(f,x)$. Hence we have $0 \leq s(f) \leq n$ for every $f : \{0,1\}^n \to \{\pm 1\}$; while not crucial, it may be helpful to consider this parameter as "low" when e.g. either $s(f) \leq (\log n)^{O(1)}$ or $s(f) \leq n^{o(1)}$ (note that both these notions of "low" are robust up to polynomial factors).

A well known conjecture, sometimes referred to as the "sensitivity conjecture," states that every smooth Boolean function is computed by a low degree real polynomial, specifically of degree polynomial in its sensitivity. This conjecture was first posed in the form of a question by Nisan [20] and Nisan and Szegedy [19] but is now (we feel) widely believed to be true:

▶ **Conjecture 1.1** ([20, 19]). *There exists a constant c such that every Boolean function f is computed by a polynomial of degree* $\deg(f) \leq s(f)^c$.

Despite significant effort ([17, 1, 2, 3, 4]) the best upper bound on degree in terms of sensitivity is exponential. Recently several consequences of Conjecture 1.1, e.g. that every f has a *formula* of depth at most $\mathrm{poly}(s(f))$, have been unconditionally established in [11]. Nisan and Szegedy proved the converse, that every Boolean function satisfies $s(f) = O(\deg(f)^2)$.

In this work, we make progress on Conjecture 1.1 by showing that functions with low max sensitivity are very well *approximated* (in ℓ_2) by low-degree polynomials. We exponentially improve the folklore $O(s/\epsilon)$ degree bound (which follows from average sensitivity and Markov's inequality) by replacing the $1/\epsilon$ error dependence with $\mathrm{poly}\log(1/\epsilon)$. The following is our main result:[1]

▶ **Theorem 1.2.** *For any Boolean function $f : \{0,1\}^n \to \{\pm 1\}$ and any $\epsilon > 0$, there exists a polynomial $p : \{0,1\}^n \to \mathbb{R}$ with $\deg(p) \leq O(s(f) \cdot (\log(1/\epsilon))^3)$ such that $\mathbb{E}_{x \in \{0,1\}^n}[|p(x) - f(x)|^2] \leq \epsilon$.*

En route to proving this result, we make two related contributions which we believe are interesting in themselves:

- Formulating a robust variant of the sensitivity conjecture (which would generalize Theorem 1.2).
- Defining and analyzing some natural graph-theoretic complexity measures, essential to our proof and which we believe may hold the key to progress on the original and robust sensitivity conjectures.

1.1 A robust variant of the sensitivity conjecture

A remarkable series of developments, starting with [20], showed that real polynomial degree is an extremely versatile complexity measure: it is polynomially related to many other complexity measures for Boolean functions, including PRAM complexity, block sensitivity, certificate complexity, deterministic/randomized/quantum decision tree depth, and approximating polynomial degree (see [6, 15] for details on many of these relationships). Arguably the one natural complexity measure that has defied inclusion in this equivalence class is

[1] In a subsequent version of this work [12] the exponent "3" in Theorem 1.2 is improved to 1, and it is shown that any further improvement to an exponent strictly less than 1 implies Conjecture 1.1.

sensitivity. Thus, there are many equivalent formulations of Conjecture 1.1; indeed, Nisan's original formulation was in terms of sensitivity versus block sensitivity [20].

Even though progress on it has been slow, over the years Conjecture 1.1 has become a well-known open question in the study of Boolean functions. It is natural to ask *why* this is an important question: will a better understanding of sensitivity lead to new insights into Boolean functions that have eluded us so far? Is sensitivity qualitatively different from the other concrete complexity measures that we already understand?

We believe that the answer is yes, and in this paper we make the case that Conjecture 1.1 is just the (extremal) tip of the iceberg: it hints at deep connections between the *combinatorial* structure of a Boolean function f, as captured by the graph G_f of its sensitive edges in the hypercube, and the *analytic* structure, as captured by its Fourier expansion. This connection is already the subject of some of the key results in the analysis of Boolean functions, such as [16, 9], as well as important open problems like the "entropy-influence" conjecture [10] and its many consequences.

Given any Boolean function f, we conjecture a connection between the distribution of the sensitivity of a random vertex in $\{0,1\}^n$ and the distribution of f's Fourier mass. This conjecture, which is an important motivation for the study in this paper, is stated informally below:

Robust Sensitivity Conjecture (Informal Statement): *If **most** inputs to a Boolean function f have low sensitivity, then **most** of the Fourier mass of f is concentrated on small subsets.*

Replacing both occurrences of **most** by *all* we recover Conjecture 1.1, and hence the statement may be viewed as a robust formulation of the sensitivity conjecture. Theorem 1.2 corresponds to replacing the first **most** by *all*. There are natural classes of functions which do not have low max sensitivity, but for which most vertices have low sensitivity; the robust sensitivity conjecture is relevant to these functions while the original sensitivity conjecture is not. (A prominent example of such a class is AC^0, for which the results of [18] establish a weak version of the assumption (that most inputs have low sensitivity) and the results of [18, 26] establish a strong version of the conclusion (Fourier concentration).)

In order to formulate a precise statement, for a given Boolean function $f : \{0,1\}^n \to \{\pm 1\}$ we consider the random experiment which samples from the following two distributions:

1. The Sensitivity distribution: sample a uniform random vertex $\boldsymbol{x} \in \{0,1\}^n$ and let $\boldsymbol{s} = s(f, \boldsymbol{x})$.
2. The Fourier distribution: sample a subset $\mathbf{T} \subset [n]$ with probability $\hat{f}(\mathbf{T})^2$ and let $\boldsymbol{d} = |\mathbf{T}|$.

We conjecture a close relation between the k^{th} moments of these random variables:

▶ **Conjecture 1.3** (Robust Sensitivity Conjecture)**.** *For all Boolean functions f and for all integers $k \geq 1$, there is a constant a_k such that $\mathbb{E}[\boldsymbol{d}^k] \leq a_k \, \mathbb{E}[\boldsymbol{s}^k]$.*

The key here is that there is no dependence on n. To see the connection with the informal statement above, if a function has low sensitivity for most $x \in \{0,1\}^n$, then it must have bounded k^{th} sensitivity moments for fairly large k; in such a case, Conjecture 1.3 implies a strong Fourier concentration bound by Markov's inequality. The classical Fourier expansion for average sensitivity tells us that when $k = 1$, $\mathbb{E}[\boldsymbol{s}] = \mathbb{E}[\boldsymbol{d}]$. It is also known that $\mathbb{E}[\boldsymbol{s}^2] = \mathbb{E}[\boldsymbol{d}^2]$ (see e.g. [7, Lemma 3.5]), but equality does not hold for $k \geq 3$. Conjecture 1.3 states that if we allow constant factors depending on k, then one direction still holds.

It is clear that Conjecture 1.3 (with a_k a not-too-rapidly-growing function of k) is a strengthening of our Theorem 1.2. To see its relation to Conjecture 1.1 observe that

Conjecture 1.1 implies that for $k \to \infty$, $\mathbb{E}[\boldsymbol{d}^k] \leq a^k(\mathbb{E}[\boldsymbol{s}^k])^b$ for constants a, b. On the other hand, via Markov's inequality, Conjecture 1.3 only guarantees Fourier concentration rather than small degree for functions with small sensitivity. Thus the robust version Conjecture 1.3 seems incomparable to Conjecture 1.1.

It is possible that the reverse direction of the robust conjecture also holds: for every k there exists a'_k such that $\mathbb{E}[\boldsymbol{s}^k] \leq a'_k \mathbb{E}[\boldsymbol{d}^k]$; settling this is an intriguing open question. We note that the Nisan-Szegedy result that $s(f) \leq O(\deg(f)^2)$ implies that as $k \to \infty$ we have $\mathbb{E}[\boldsymbol{s}^k] \leq C^k \mathbb{E}[\boldsymbol{d}^k]^2$ for some constant C.

Both our proof of Theorem 1.2, and our attempts at Conjecture 1.3, follow the same general path. We apply random restrictions, which reduces these statements to analyzing some natural new graph-theoretic complexity measures of Boolean functions. These measures are relaxations of sensitivity: they look for occurrences of various subgraphs in the sensitivity graph, rather than just high degree vertices. We establish (and conjecture) connections between different graph-theoretic measures and decision tree depth (see Theorem 5.4, which relates decision tree depth and the length of "proper walks", and Conjecture 4.10, which conjectures a relation between "tree sensitivity" and decision tree depth). These connections respectively enable the proof of Theorem 1.2 and provide a simple sufficient condition implying Conjecture 1.3, which suffices to prove the conjecture for $k = 3$ and 4. We elaborate on this in the next subsection. We believe that these new complexity measures are interesting and important in their own right, and that understanding them better may lead to progress on Conjecture 1.1.

1.2 Random restrictions and graph-theoretic complexity measures

In this subsection we give a high level description of our new complexity measures and perspectives on the sensitivity graph and of how we use them to approach Conjecture 1.3 and prove Theorem 1.2. As both have the same conclusion, namely strong Fourier concentration, we describe both approaches together until they diverge. This leads to analyzing two different graph parameters (as we shall see, the stronger assumption of Theorem 1.2 allows the use of a weaker graph parameter that we can better control).

First we give a precise definition of the *sensitivity graph*: to every Boolean function f we associate a graph G_f whose vertex set is $\{0,1\}^n$ and whose edge set E consists of all edges (x, y) of the hypercube that have $f(x) \neq f(y)$. Each edge is labelled by the coordinate in $[n]$ at which x and y differ. The degree of vertex x is exactly $s(f, x)$, and the maximum degree of G_f is $s(f)$.

The starting point of our approach is to reinterpret the moments of the degree and sensitivity distributions of f in terms of its random restrictions. Let $\mathcal{R}_{k,n}$ denote the distribution over random restrictions that leave exactly k of the n variables unset and set the rest uniformly at random. We first show, in Section 3, that the k^{th} moment of the sensitivity distribution controls the probability that a random restriction $f_{\boldsymbol{\rho}}$ of f, where $\boldsymbol{\rho} \leftarrow \mathcal{R}_{k,n}$, has full sensitivity (Theorem 3.1). Similarly, moments of the Fourier distribution capture the event that $f_{\boldsymbol{\rho}}$ has full degree (Theorem 3.2).[2]

Random restrictions under sensitivity moment bounds

Via Theorems 3.1 and 3.2, Conjecture 1.3 may be rephrased as saying that if a function f has low sensitivity moments, then a random restriction $f_{\boldsymbol{\rho}}$ is unlikely to have full degree. An

[2] We note that Tal has proved a result of a similar flavor; [25, Theorem 3.2] states that strong Fourier concentration of f implies that random restrictions of f are unlikely to have high degree.

intuition supporting this statement is that the sensitivity graphs of functions with full degree should be "complex" (under some suitable complexity measure), whereas the graph of $f_{\boldsymbol{\rho}}$ is unlikely to be "complex" if f has low sensitivity moments. More precisely, the fact that G_f has no (or few) vertices of high degree suggests that structures with many sensitive edges in distinct directions will not survive a random restriction.

Some evidence for this intuition is given by Theorem 3.1, which tells us that if f has low sensitivity moments then $f_{\boldsymbol{\rho}}$ is unlikely to have full sensitivity. If full degree implied full sensitivity then we would be done, but this is false as witnessed e.g. by the three-variable majority function and by composed variants of it. (Conjecture 1.1 asserts that the gap between degree and sensitivity is at most polynomial, but of course we do not want to invoke the conjecture!) This leads us in Section 4 to consider our first relaxation of sensitivity, which we call *tree-sensitivity*. To motivate this notion, note that a vertex with sensitivity k is simply a star with k edges in the sensitivity graph. We relax the star requirement and consider all *sensitive trees*: trees of sensitive edges (i.e. edges in G_f) where every edge belongs to a *distinct* coordinate direction (as is the case, of course, for a star). Analogous to the usual notion of sensitivity, the tree sensitivity of f at x is the size of the largest sensitive tree containing x, and the tree sensitivity of f is the maximum tree sensitivity of f at any vertex.

Theorem 4.11 shows that the sensitivity moments of f control the probability that f_ρ has full tree sensitivity. Its proof crucially uses a result by Sidorenko [24] on counting homomorphisms to trees. Theorem 4.11 would immediately imply Conjecture 1.3 if every function of degree k must have tree sensitivity k. (This is easily verified for $k = 3, 4$, which, as alluded to in the previous subsection, gives Conjecture 1.3 for those values of k.) The best we can prove, though, is a tree sensitivity lower bound of $\Omega(\sqrt{k})$ (Theorem 4.9); the proof uses notions of maximality and "shifting" of sensitive trees that we believe may find further application in the study of tree sensitivity. We conjecture that full degree does imply full tree sensitivity, implying Conjecture 1.3. This is a rare example where having a precise bound between the two complexity measures (rather than a polynomial relationship) seems to be important.

Random restrictions under a max sensitivity bound

Next, we aim to prove *unconditional* moment bounds on the Fourier distribution of functions with low max sensitivity, and thereby obtain Theorem 1.2. Towards this goal, in Section 5 we relax the notion of tree sensitivity and study certain walks in the Boolean hypercube that we call *proper walks*: these are walks such that every time a coordinate direction is explored for the first time, it is along a sensitive edge. We show in Theorem 5.4 that having full decision tree depth implies the existence of a very short (length $O(n)$) proper walk containing sensitive edges along every coordinate. In Lemma 5.6, we analyze random restrictions to show that such a structure is unlikely to survive in the remaining subcube of unrestricted variables. This may be viewed as a "switching lemma for low-sensitivity functions", which again may be independently interesting (note that strictly speaking this result is not about switching from a DNF to a CNF or vice versa, but rather it upper bounds the probability that a restricted function has large decision tree depth, in the spirit of standard "switching lemmas"). It yields Theorem 1.2 via a rather straightforward argument. The analysis requires an upper bound on the maximum sensitivity because we do not know an analogue of Sidorenko's theorem for proper walks.

1.3 Some high-level perspective

An important goal of this work is to motivate a better understanding of the combinatorial structure of the sensitivity graph G_f associated with a Boolean function. In our proofs other notions suggest themselves beyond tree sensitivity and proper walks, most notably the *component dimension* of the graph, which may be viewed as a further relaxation of sensitivity. Better relating these measures to decision tree depths, as well as to each other, remains intriguing, and in our view promising, for making progress on Conjecture 1.1 and Conjecture 1.3 and for better understanding Boolean functions in general. We hope that some of the notions and techniques we introduce in this work will be of use to this goal.

Another high level perspective relates to "switching lemmas". As mentioned above, we prove here a new result of this kind, showing that under random restrictions low sensitivity functions have low decision tree depth with high probability. The classical switching lemma shows the same for small width DNF (or CNF) formulas (and hence for AC^0 circuits as well). Our proof is quite different than the standard proofs, as it is essentially based on the combinatorial parameters of the sensitivity graph. Let us relate the assumptions of both switching lemmas. On the one hand, by the sensitivity Conjecture 1.1 (which we can't use, and want to prove), low sensitivity should imply low degree and hence low decision tree depth and small DNF width. On the other hand, small DNF width (indeed small, shallow circuits) imply (by [18]) low *average* sensitivity, which is roughly the assumption of the robust sensitivity Conjecture 1.3. As it turns out, we can use our combinatorial proof of our switching lemma to derive a somewhat weaker form of the original switching lemma, and also show that the same combinatorial assumption (relating tree sensitivity to decision tree depth) which implies Conjecture 1.3 would yield a nearly tight form of the original switching lemma. This lends further motivation to the study of these graph parameters.

Another conjecture formalizing the maxim that *low sensitivity implies Fourier concentration* is the celebrated Entropy-Influence conjecture of Freidgut and Kalai [10] which posits the existance of a universal constant C such that $H(\mathbf{T}) \leq C\,\mathbb{E}[\boldsymbol{s}]$ where $H(.)$ denotes the entropy function of a random variable.[3] The conjecture states that functions with low sensitivity on average (measured by $\mathbb{E}[\boldsymbol{s}] = \mathbb{E}[\boldsymbol{d}]$) have their Fourier spectrum concentrated on a few coefficients, so that the entropy of the Fourier distribution is low. However, unlike in Conjecture 1.3 the degree of those coefficients does not enter the picture.

Organization

We present some standard preliminaries and notation in Section 2. Section 3 proves Theorems 3.1 and 3.2 which show that degree and sensitivity moments govern the degree and sensitivity respectively of random restrictions. In Section 4 we study tree sensitivity. Section 4.1 relates it to other complexity measures, while Section 4.2 shows how the tree sensitivity of a random restriction is governed by sensitivity moments. We explore some consequences of these results in Section 4.3. Section 5 studies proper walks, and shows how to construct short proper walks. In Section 5.1, we use proper walks to analyze random restrictions of low-sensitivity functions and prove Theorem 1.2. Section 5 uses results from Section 4.1 but is independent of the rest of Section 4.

[3] Recall that the entropy $H(\mathbf{T})$ of the random variable $\mathbf{T}$ is $H(\mathbf{T}) = \sum_{T \subseteq [n]} \Pr[\mathbf{T} = T] \log_2 \frac{1}{\Pr[\mathbf{T}=T]}$.

2 Preliminaries

The Fourier distribution. Let $f : \{0,1\}^n \to \{\pm 1\}$ be a Boolean function. We define the usual inner product on the space of such functions by $\langle f, g\rangle = \mathbb{E}_{\boldsymbol{x}\leftarrow\{0,1\}^n}[f(\boldsymbol{x})g(\boldsymbol{x})]$. For $S \subseteq [n]$ the parity function χ_S is $\chi_S(x) = (-1)^{\sum_{i\in S} x_i}$. The Fourier expansion of f is given by $f(x) = \sum_{S\subset[n]} \hat{f}(S)\chi_S(x)$, where $\hat{f}(S) = \langle f, \chi_S\rangle$. By Parseval's identity we have $\sum_{S\subseteq[n]} \hat{f}(S)^2 = 1$. This allows us to view any Boolean function f as inducing a probability distribution $\mathcal{D}_f$ on subsets $S \subseteq [n]$, given by $\Pr_{\mathbf{R}\leftarrow\mathcal{D}_f}[\mathbf{R} = S] = \hat{f}(S)^2$. We refer to this as the *Fourier distribution*. We define $\mathrm{supp}(f) \subseteq 2^{[n]}$ as $\mathrm{supp}(f) = \{S \subseteq [n] : \hat{f}(S)^2 \neq 0\}$. The Fourier expansion of f can be viewed as expressing S as a multilinear polynomial in $x_1, \ldots, x_n$, so that $\deg(f) = \max_{S\in\mathrm{supp}(f)} |S|$. Viewing $\mathcal{D}_f$ as a probability distribution on $2^{[n]}$, we define the following quantities which we refer to as "influence moments" of f:

$$\mathbb{I}^{\mathrm{k}}[f] = \mathop{\mathbb{E}}_{\mathbf{R}\leftarrow\mathcal{D}_f}\left[|\mathbf{R}|^k\right] = \sum_S \hat{f}(S)^2|S|^k, \tag{1}$$

$$\mathbb{I}^{\underline{\mathrm{k}}}[f] = \mathop{\mathbb{E}}_{\mathbf{R}\leftarrow\mathcal{D}_f}\left[\prod_{i=0}^{k-1}(|\mathbf{R}| - i)\right] = \sum_{|S|\geq k} \hat{f}(S)^2 \prod_{i=0}^{k-1}(|S| - i). \tag{2}$$

We write $\deg_\epsilon(f)$ to denote the minimum k such that $\sum_{S\subseteq[n];|S|\geq k} \hat{f}(S)^2 \leq \epsilon$. It is well known that $\deg_\epsilon(f) \leq k$ implies the existence of a degree k polynomial g such that $\mathbb{E}_{\boldsymbol{x}}[(f(\boldsymbol{x}) - g(\boldsymbol{x}))^2] \leq \epsilon$; g is obtained by truncating the Fourier expansion of f to level k.

The sensitivity distribution. We use $d(\cdot,\cdot)$ to denote Hamming distance on $\{0,1\}^n$. The n-dimensional hypercube H_n is the graph with vertex set $V = \{0,1\}^n$ and $\{x, y\} \in E$ if $d(x, y) = 1$. For $x \in \{0,1\}^n$, let $N(x)$ denote its neighborhood in H_n. As described in Section 1, the *sensitivity* of a function f at point x is defined as $s(f, x) = |\{y \in N(x) : f(x) \neq f(y)\}|$, and the (worst-case) sensitivity of f, denoted $s(f)$, is defined as $s(f) = \max_{x\in\{0,1\}^n} s(f, x)$. Analogous to (1) and (2), we define the quantities $s^k(f)$ and $s^{\underline{k}}(f)$ which we refer to as "sensitivity moments" of f:

$$s^k(f) = \mathop{\mathbb{E}}_{\boldsymbol{x}\leftarrow\{0,1\}^n}\left[s(f,\boldsymbol{x})^k\right], \qquad s^{\underline{k}}(f) = \mathop{\mathbb{E}}_{\boldsymbol{x}\leftarrow\{0,1\}^n}\left[\prod_{i=0}^{k-1}(s(f,\boldsymbol{x}) - i)\right]. \tag{3}$$

With this notation, we can restate Conjecture 1.3 (with a small modification) as

▶ **Conjecture** (Conjecture 1.3 restated). *For every k, there exists constants a_k, b_k such that $\mathbb{I}^k(f) \leq a_k s^k(f) + b_k$.*

The reason for the additive constant b_k is that for all non-negative integers x, we have $\prod_{i=0}^{k-1}(x - i) \leq x^k \leq e^k \prod_{i=0}^{k-1}(x - i) + k^k$. Hence allowing the additive factor lets us freely interchange $\mathbb{I}^k$ with $\mathbb{I}^{\underline{k}}$ and s^k with $s^{\underline{k}}$ in the statement of the Conjecture. We note that $\mathbb{I}^1[f] = \mathbb{I}^{\underline{1}}[f] = s^1(f) = s^{\underline{1}}(f)$, and as stated earlier it is not difficult to show that $\mathbb{I}^2[f] = s^2(f)$ (see e.g. Lemma 3.5 of [7]). However, in general $\mathbb{I}^k(f) \neq s^k(f)$ for $k \geq 3$ (as witnessed, for example, by the AND function).

Some other complexity measures. We define $\dim(f)$ to be the number of variables that f depends on and $\mathrm{dt}(f)$ to be the smallest depth of a deterministic decision tree computing f. In particular $f : \{0,1\}^n \to \{\pm 1\}$ has $\dim(f) = n$ iff f is sensitive to every co-ordinate, and has $\mathrm{dt}(f) = n$ iff f is evasive. It is easy to see that $\deg(f) \leq \mathrm{dt}(f) \leq \dim(f)$ and $s(f) \leq \mathrm{dt}(f)$.

3 Random restrictions and moments of degree and sensitivity

We write $\mathcal{R}_{k,n}$ to denote the set of all restrictions that leave exactly k variables live (unset) out of n. A restriction $\rho \in \mathcal{R}_{k,n}$ is viewed as a string in $\{0,1,\star\}^n$ where $\rho_i = \star$ for exactly the k live variables. We denote the set of live variables by $L(\rho)$, and we use $f_\rho : \{0,1\}^{L(\rho)} \to \{\pm 1\}$ to denote the resulting restricted function. We use $C(\rho) \subseteq \{0,1\}^n$ to denote the subcube consisting of all possible assignments to variables in $L(\rho)$. We sometimes refer to "a random restriction $\boldsymbol{\rho} \leftarrow \mathcal{R}_{k,n}$" to indicate that $\boldsymbol{\rho}$ is selected uniformly at random from $\mathcal{R}_{k,n}$.

A random restriction $\boldsymbol{\rho} \leftarrow \mathcal{R}_{k,n}$ can be chosen by first picking a set $\mathbf{K} \subset [n]$ of k co-ordinates to set to $\star$ and then picking $\boldsymbol{\rho}_{\overline{\mathbf{K}}} \in \{0,1\}^{[n]\setminus\mathbf{K}}$ uniformly at random. Often we will pick both $\boldsymbol{x} \in \{0,1\}^n$ and $\mathbf{K} \subset [n]$ of size k independently and uniformly at random. This is equivalent to sampling a random restriction $\boldsymbol{\rho}$ and a random point $\boldsymbol{y}$ within the subcube $C(\rho)$.

The following two theorems show that $\mathbb{I}^{\underline{k}}[f]$ captures the degree of f_ρ, whereas $s^{\underline{k}}(f)$ captures its sensitivity.

▶ **Theorem 3.1.** *Let $f : \{0,1\}^n \to \{\pm 1\}$, $\rho \leftarrow \mathcal{R}_{k,n}$, and $1 \le j \le k$. Then*

$$\frac{s^{\underline{j}}(f)}{n^j} \approx \frac{s^{\underline{j}}(f)}{\prod_{i=0}^{j-1}(n-i)} \le \Pr_{\boldsymbol{\rho}\leftarrow\mathcal{R}_{k,n}}[s(f_{\boldsymbol{\rho}}) \ge j] \le \frac{2^k s^{\underline{j}}(f)\binom{k}{j}}{\prod_{i=0}^{j-1}(n-i)} \approx \frac{2^k s^{\underline{j}}(f)\binom{k}{j}}{n^j}. \tag{4}$$

Proof. Consider the bipartite graph in which the vertices X on the left are all j-edge stars S in G_f, the vertices Y on the right are all restrictions $\rho \in \mathcal{R}_{k,n}$, and an edge connects S and ρ if the star S lies in the subcube $C(\rho)$ specified by the restriction ρ. The desired probability $\Pr_{\rho\in\mathcal{R}_{k,n}}[s(f_\rho) \ge j]$ is the fraction of nodes in Y that are incident to at least one edge.

The number of nodes on the left is equal to

$$|X| = \sum_{x\in\{0,1\}^n} \binom{s(f,x)}{j} = \frac{2^n s^{\underline{j}}(f)}{j!}.$$

The degree of each node S on the left is exactly $\binom{n-j}{k-j}$, since if S is adjacent to ρ then j of the k elements of $L(\rho)$ must correspond to the j edge coordinates of S and the other $k-j$ elements of $L(\rho)$ can be any of the $n-j$ remaining coordinates (note that the non-$\star$ coordinates of ρ are completely determined by S). On the right, a restriction $\rho \in \mathcal{R}_{k,n}$ is specified by a set $L(\rho)$ of k live co-ordinates where $\rho_i = \star$, and a value $\rho_i \in \{0,1\}$ for the other coordinates, so $|Y| = |\mathcal{R}_{k,n}| = \binom{n}{k}2^{n-k}$. We thus have

$$\Pr_{\boldsymbol{\rho}\leftarrow\mathcal{R}_{k,n}}[s(f_{\boldsymbol{\rho}}) \ge j] \le \frac{\text{total \# of edges into } Y}{|Y|} = \frac{\left(\frac{2^n s^{\underline{j}}(f)}{j!}\right)\cdot\binom{n-j}{k-j}}{\binom{n}{k}2^{n-k}} = \frac{2^k s^{\underline{j}}(f)\binom{k}{j}}{\prod_{i=0}^{j-1}(n-i)}.$$

For the lower bound, in order for S to lie in $C(\rho)$ the root of S must belong to $C(\rho)$ (2^k choices) and all edges of S must correspond to elements of $L(\rho)$ ($\binom{k}{j}$ choices), so the maximum degree of any $\rho \in Y$ is $2^k\binom{k}{j}$. Hence we have

$$\Pr_{\boldsymbol{\rho}\leftarrow\mathcal{R}_{k,n}}[s(f_{\boldsymbol{\rho}}) \ge j] \ge \frac{\frac{\text{(total \# of edges into } Y)}{\text{(max degree of any } \rho\in Y)}}{|Y|} = \frac{\left(\frac{2^n s^{(j)}(f)}{j!}\right)\cdot\binom{n-j}{k-j}}{2^k\binom{k}{j}\cdot\binom{n}{k}2^{n-k}} = \frac{s^{\underline{j}}(f)}{\prod_{i=0}^{j-1}(n-i)}.$$

◀

▶ **Theorem 3.2.** [4] *Let $f : \{0,1\}^n \to \{\pm 1\}$ and $\rho \leftarrow \mathcal{R}_{k,n}$. Then*

$$\frac{\mathbb{I}^{k}(f)}{n^k} \approx \frac{\mathbb{I}^{k}(f)}{\prod_{i=0}^{k-1}(n-i)} \leq \Pr_{\boldsymbol{\rho} \leftarrow \mathcal{R}_{k,n}}[\deg(f_{\boldsymbol{\rho}}) = k] \leq \frac{2^{2k-2}\,\mathbb{I}^{k}(f)}{\prod_{i=0}^{k-1}(n-i)} \approx \frac{2^{2k-2}\,\mathbb{I}^{k}(f)}{n^k}. \tag{5}$$

Proof. We first fix $K \subseteq [n]$ and consider the restricted function $f_{\boldsymbol{\rho}}$ that results from a random choice of $\boldsymbol{y} = \boldsymbol{\rho}_{\bar{K}} \in \{0,1\}^{[n]\setminus K}$. The degree k Fourier coefficient of $f_{\boldsymbol{\rho}}$ equals $\hat{f}_{\boldsymbol{\rho}}(K)$ and is given by

$$\hat{f}_{\boldsymbol{\rho}}(K) = \sum_{S \subset [n]\setminus K} \hat{f}(S \cup K)\chi_S(\boldsymbol{y}).$$

Hence we have

$$\mathop{\mathbb{E}}_{\boldsymbol{y}}[\hat{f}_{\boldsymbol{\rho}}(K)^2] = \sum_{S \subset [n]\setminus K} \hat{f}(S \cup K)^2,$$

and hence over a random choice of $\mathbf{K}$, we have

$$\mathop{\mathbb{E}}_{\boldsymbol{\rho}}[\hat{f}_{\boldsymbol{\rho}}(\mathbf{K})^2] = \sum_{S \subset [n]} \mathop{\mathbb{E}}_{\boldsymbol{\rho}}[\mathbf{1}(\mathbf{K} \subseteq S)]\hat{f}(S)^2 = \sum_{S \subset [n]} \frac{\prod_{i=0}^{k-1}(|S|-i)}{\prod_{i=0}^{k-1}(n-i)}\hat{f}(S)^2 = \frac{\mathbb{I}^{k}[f]}{\prod_{i=0}^{k-1}(n-i)}. \tag{6}$$

Note that $\deg(f_{\boldsymbol{\rho}}) = k$ iff $\widehat{f_{\boldsymbol{\rho}}}(\mathbf{K})^2 \neq 0$. Further, when it is non-zero $\widehat{f_{\boldsymbol{\rho}}}(\mathbf{K})^2$ lies in the range $[2^{-(2k-2)}, 1]$, since a non-zero Fourier coefficient in a k-variable Boolean function has magnitude at least 2^{-k+1}. Hence we have

$$2^{-2k+2}\Pr_{\boldsymbol{\rho}}[\widehat{f_{\boldsymbol{\rho}}}(\mathbf{K})^2 \neq 0] \leq \mathop{\mathbb{E}}_{\boldsymbol{\rho}}[\widehat{f_{\boldsymbol{\rho}}}(\mathbf{K})^2] \leq \Pr_{\boldsymbol{\rho}}[\widehat{f_{\boldsymbol{\rho}}}(\mathbf{K})^2 \neq 0] \tag{7}$$

which gives the desired bound when plugged into Equation (6). ◀

Conjecture 1.3 revisited: An easy adaptation of the Theorem 3.2 argument gives bounds on $\Pr_{\boldsymbol{\rho} \leftarrow \mathcal{R}_{k,n}}[\deg(f_{\boldsymbol{\rho}}) \geq j]$. Given these bounds, Conjecture 1.3 implies that for any $j \leq k$,

$$\Pr_{\boldsymbol{\rho} \leftarrow \mathcal{R}_{k,n}}[\deg(f_{\boldsymbol{\rho}}) \geq j] \leq a_k \Pr_{\boldsymbol{\rho} \leftarrow \mathcal{R}_{k,n}}[s(f_{\boldsymbol{\rho}}) \geq j] + o_n(1).$$

Indeed, by specifying the $o_n(1)$ term, we can get a reformulation of Conjecture 1.3. This formulation has an intuitive interpretation: *gap examples exhibiting low sensitivity but high degree are not robust to random restrictions.* Currently, we do not know how to upper bound $\deg(f)$ by a polynomial in $s(f)$, indeed we do know of functions f where $\deg(f) \geq s(f)^2$. But the Conjecture implies that if we hit any function f with a random restriction, the probability that the restriction has large degree can be bounded by the probability that it has large sensitivity. Thus the conjecture predicts that these gaps do not survive random restrictions in a rather strong sense.

Implications for AC^0: For functions with small AC^0 circuits, a sequence of celebrated results culminating in the work of Håstad [14] gives upper bounds on $\Pr[\mathrm{dt}(f_{\boldsymbol{\rho}}) \geq j]$. Since $\Pr[\mathrm{dt}(f_{\boldsymbol{\rho}}) \geq j] \geq \Pr[\deg(f_{\boldsymbol{\rho}}) \geq j]$, we can plug these bounds into Theorem 3.2 to get upper

[4] The upper bound in the following theorem is essentially equivalent to Theorem 3.2 of [25], while the lower bound is analogous to [18]. The only difference is in the family of restrictions.

bounds on the Fourier moments, and derive a statement analogous to [18, Lemma 7], [26, Theorem 1.1] on the Fourier concentration of functions in AC^0.

Similarly $\Pr[\mathrm{dt}(f_{\boldsymbol{\rho}}) \geq j] \geq \Pr[s(f_{\boldsymbol{\rho}}) \geq j]$, so via this approach Theorem 3.1 gives upper bounds on the sensitivity moments, and hence sensitivity tail bounds for functions computed by small AC^0 circuits. This can be viewed as an extension of [18, Lemma 12], which bounds the average sensitivity (first moment) of such functions. For depth 2 circuits, such tail bounds are implied by the satisfiability coding lemma [21], but we believe these are the first such bounds for depth 3 and higher. As this is not the focus of our current work, we leave the details to the interested reader.

4 Tree sensitivity

In this section we study the occurrence of trees of various types in the sensitivity graph G_f, by defining a complexity measure called tree sensitivity. We study its relation to other complexity measures like decision tree depth.

▶ **Definition 4.1.** A set $S \subseteq \{0,1\}^n$ *induces a sensitive tree* T in G_f if (i) the points in S induce the (non-trivial) tree T in the Boolean hypercube; (ii) every edge induced by S is a sensitive edge for f, i.e. belongs to $E(G_f)$; and (iii) each induced edge belongs to a distinct co-ordinate direction.

Given a fixed function f, a sensitive tree T is completely specified by the set $V(T)$ of its vertices. We can think of each edge $e \in E(T)$ as being labelled by the coordinate $\ell(e) \in [n]$ along which f is sensitive, so every edge has a distinct label. Let $\ell(T)$ denote the set of all edge labels that occur in T. We refer to $|\ell(T)|$ as the *size* of T, and observe that it lies in $\{1, \ldots, n\}$. We note that $|V(T)| = |\ell(T)| + 1$ by the tree property. Further, any two vertices in $V(T)$ differ on a subset of coordinates in $\ell(T)$. Hence the set $V(T)$ lies in a subcube spanned by coordinates in $\ell(T)$, and all points in $V(T)$ agree on all the coordinates in $\overline{\ell(T)} \stackrel{\text{def}}{=} [n] \setminus \ell(T)$.

▶ **Definition 4.2.** For $x \in \{0,1\}^n$, the *tree-sensitivity of f at x*, denoted $\mathrm{ts}(f,x)$, is the maximum of $|\ell(T)|$ over all sensitive trees T such that $x \in V(T)$. We define the tree-sensitivity of f as $\mathrm{ts}(f) = \max_{x \in \{0,1\}^n} \mathrm{ts}(f,x)$.

Note that a vertex and all its sensitive neighbors induce a sensitive tree (which is a star). Thus one can view tree-sensitivity as a generalization of sensitivity, and hence we have that $\mathrm{ts}(f) \geq \mathrm{s}(f)$. Lemma A.1 will show that $\mathrm{ts}(f)$ can in fact be exponentially larger than both $\mathrm{s}(f)$ and $\mathrm{dt}(f)$ (the decision tree depth of f), and thus it cannot be upper bounded by some polynomial in standard measures like decision tree depth, degree, or block sensitivity. However, Theorem 4.9, which we prove in the next subsection, gives a polynomial lower bound.

4.1 Tree sensitivity and decision tree depth

A sensitive tree T is *maximal* if there does not exist any sensitive tree T' with $V(T) \subsetneq V(T')$. In this subsection we study maximal sensitive trees using a "shifting" technique, introduce the notion of an "orchard" (a highly symmetric configuration of isomorphic sensitive trees that have been shifted in all possible ways along their insensitive coordinates), and use these notions to prove Theorem 4.9, which lower bounds tree sensitivity by square root of decision tree depth.

The *support* of a vector $v \in \{0,1\}^n$, denoted $\mathrm{supp}(v)$, is the set $\{i \in [n] : v_i = 1\}$. For $x, v \in \{0,1\}^n$, $x \oplus v$ denotes the coordinatewise xor. Given a set $S \subseteq \{0,1\}^n$, let $S \oplus v = \{x \oplus v : x \in S\}$.

▶ **Definition 4.3.** Let v be a vector supported on $\overline{\ell(T)}$ where T is a sensitive tree in G_f. We say that T can be *shifted* by v if $f(x) = f(x \oplus v)$ for all $x \in V(T)$.

If T can be shifted by v then $V(T) \oplus v$ also induces a sensitive tree which we denote by $T \oplus v$. Mapping x to $x \oplus v$ gives an isomorphism between T and $T \oplus v$ which preserves both adjacency and edge labels, and in particular we have $\ell(T \oplus v) = \ell(T)$.

We have the following characterization of maximality (both directions follow easily from the definitions of maximality and of shifting by the unit basis vector e_i):

▶ **Lemma 4.4.** *A sensitive tree T is maximal if and only if it can be shifted by e_i for all $i \in \overline{\ell(T)}$ (equivalently, if none of the vertices in $V(T)$ is sensitive to any coordinate in $\overline{\ell(T)}$).*

The notion of maximality allows for a "win-win" analysis of sensitive trees: for each co-ordinate $i \in \overline{\ell(T)}$, we can either increase the size of the tree by adding an edge in direction i, or we can shift by e_i to get an isomorphic copy of the tree. Repeating this naturally leads to the following definition.

▶ **Definition 4.5.** Let T be a sensitive tree that can be shifted by every v supported on $\overline{\ell(T)}$. We refer to the set of all such trees $F = \{T \oplus v\}$ as an *orchard*, and we say that T belongs to the orchard F.

An orchard guarantees the existence of $2^{n-\ell(T)}$ trees that are isomorphic to T in G_f. It is *a priori* unclear that orchards exist in G_f. The following simple but key lemma proves their existence.

▶ **Lemma 4.6.** *Let T be a sensitive tree. Either T belongs to an orchard, or there exists a shift $T \oplus v$ of T which is not maximal.*

Proof. Assume the tree T does not belong to an orchard. Pick the smallest weight vector v' supported on $\overline{\ell(T)}$ such that T cannot be shifted by v' (if there is more than one such vector any one will do). Since T can trivially be shifted by 0^n, we have $\mathrm{wt}(v') \geq 1$. Pick any co-ordinate $i \in \mathrm{supp}(v')$, and let $v = v' \oplus e_i$ so that $\mathrm{wt}(v) = \mathrm{wt}(v') - 1$. By our choice of v', T can be shifted by v, but not by $v' = v \oplus e_i$. This implies that there exists $x \in V(T)$ so that $f(x) = f(x \oplus v) \neq f(x \oplus v')$, hence $T \oplus v$ is not maximal. ◀

This lemma directly implies the existence of orchards for every G_f:

▶ **Corollary 4.7.** *Every sensitive tree T where $|\ell(T)| = \mathrm{ts}(f)$ belongs to an orchard.*

The lemma also gives the following intersection property for orchards. Since any two trees in an orchard F are isomorphic, we can define $\ell(F) = \ell(T)$ to be the set of edge labels for any tree $T \in F$.

▶ **Lemma 4.8.** *Let F_1 and F_2 be orchards. Then $\ell(F_1) \cap \ell(F_2) \neq \emptyset$.*

Proof. Assume for contradiction that $\ell(F_1)$ and $\ell(F_2)$ are disjoint. We choose trees $T_1 \in F_1$ and $T_2 \in F_2$, and $x \in V(T_1), y \in V(T_2)$ such that $f(x) = 1$ and $f(y) = -1$. Now define $z \in \{0,1\}^n$ where z_i equals x_i if $i \in \ell(T_1)$ and z_i equals y_i otherwise. Since z agrees with x on $\ell(T_1) = \ell(F_1)$, it can be obtained by shifting x by $z \oplus x$ which is supported on $\overline{\ell(T_1)}$. Since T_1 belongs to an orchard, we get $f(z) = f(x) = 1$. However, we also have that $z_i = y_i$ for all $i \in \ell(T_2)$. Hence by similar reasoning, $f(z) = f(y) = -1$, which is a contradiction. ◀

We use this intersection property to lower bound tree sensitivity in terms of decision tree depth, via an argument similar to other upper bounds on $\mathrm{dt}(f)$ (such as the well known [5, 27, 13] quadratic upper bound on $\mathrm{dt}(f)$ in terms of certificate complexity).

▶ **Theorem 4.9.** *For any Boolean function $f : \{0,1\}^n \to \{\pm 1\}$, we have $\mathrm{ts}(f) \geq \sqrt{2\,\mathrm{dt}(f)} - 1$.*

Proof. We construct a decision tree for f by iterating the following step until we are left with a constant function at each leaf: at the current node in the decision tree, pick the largest sensitive tree T in the (restricted) function and read all the variables in $\ell(T)$.

Let k be the largest number of iterations before we terminate, taken over all paths in the decision tree. Fix a path that achieves k iterations and let f_i be the restriction of f that is obtained, at the end of the i-th iteration (and let $f_0 = f$). We claim that $\mathrm{ts}(f_i) \leq \mathrm{ts}(f) - i$. Note that if f_i is not constant then $\mathrm{ts}(f_i) \geq 1$, hence this claim implies that $k \leq \mathrm{ts}(f)$.

It suffices to prove the case $i = 1$, since we can then apply the same argument repeatedly. Consider all trees in $f_0 = f$ of size $\mathrm{ts}(f)$. Each of them occurs in an orchard by Corollary 4.7 and by Lemma 4.8 any two of them share at least one variable. Hence when we read all the variables in some tree T, we restrict at least one variable in every tree of size $\mathrm{ts}(f)$, reducing the size by at least 1. The size of the other trees cannot increase after restriction, since G_{f_1} is an induced subgraph of G_f. Hence all the sensitive trees in f_1 have size at most $\mathrm{ts}(f) - 1$.

It follows that overall we can bound the depth of the resulting decision tree by

$$\mathrm{dt}(f) \leq \sum_{i=1}^{k} \mathrm{ts}(f_{i-1}) \leq \sum_{i=1}^{k} (\mathrm{ts}(f) - (i-1)) \leq \frac{\mathrm{ts}(f)(\mathrm{ts}(f)+1)}{2}. \qquad \blacktriangleleft$$

It is natural to ask whether $\mathrm{ts}(f)$ is polynomially related to $\mathrm{dt}(f)$ and other standard complexity measures. Lemma A.1 in Appendix A gives an example of a function on n variables where $\mathrm{dt}(f) = \log(n+1)$ whereas $\mathrm{ts}(f) = n$. In the other direction, it is likely that the bound in Theorem 4.9 can be improved further. We conjecture that the following bound should hold:

▶ **Conjecture 4.10.** *For any Boolean function $f : \{0,1\}^n \to \{\pm 1\}$, we have $\mathrm{ts}(f) \geq \mathrm{dt}(f)$.*

In addition to being a natural question by itself, we will show in Section 4.3 that Conjecture 4.10 would have interesting consequences via the switching lemma in Section 4.2.

4.2 Tree Sensitivity under Random Restrictions

In this subsection we show that the probability of a random restriction of f having large tree sensitivity is both upper and lower bounded by suitable sensitivity moments of f.

▶ **Theorem 4.11.** *Let $f : \{0,1\}^n \to \{\pm 1\}$, $\boldsymbol{\rho} \sim \mathcal{R}_{k,n}$ and $1 \leq j \leq k$. Then we have*

$$\frac{s^{\underline{j}}(f)}{n^j} \approx \frac{s^{\underline{j}}(f)}{\prod_{i=0}^{j-1}(n-i)} \leq \Pr_{\boldsymbol{\rho} \in \mathcal{R}_{k,n}}[\mathrm{ts}(f_{\boldsymbol{\rho}}) \geq j] \leq \frac{(2k)^{2k} s^j(f)}{\prod_{i=0}^{j-1}(n-i)} \approx \frac{(2k)^{2k} s^j(f)}{n^j}.$$

The lower bound follows from the fact that $\mathrm{ts}(f) \geq s(f)$ and Theorem 3.1. The key ingedient in the upper bound is Sidorenko's theorem [24], which bounds the number of homomorphisms from a graph G to a tree T with j edges in terms of the j^{th} degree moment of G. For a formal statement of Sidorenko's theorems, we refer the reader to [24, 8]. Below, we state the result we will use in our language. We also present an elegant proof due to Yuval Peres which seems considerably simpler than the known proofs of Sidorenko's theorem (though the lemma follows directly from that theorem).

▶ **Lemma 4.12** ([22]). *Let $\mathcal{S}_j$ denote the set of sensitive trees of size j in G_f. Then we have that*

$$|\mathcal{S}_j| \leq j! \sum_{x \in \{0,1\}^n} s(f,x)^j.$$

Proof. We consider the set $\mathcal{T}$ of all rooted unlabelled trees with j edges. For each tree $t \in \mathcal{T}$ we define a labelling of its vertices as follows: each tree $t \in \mathcal{T}$ has the vertex set $V = \{0, \ldots, j\}$ where 0 is the root. The adjacency structure is specified by a parent function $p_t : \{1, \ldots, j\} \to \{0, \ldots, j-1\}$ where $p_t(i) < i$ is the parent of vertex i. The tree t is completely specified by the function p_t, and hence $|\mathcal{T}| \leq j!$. For a given $t \in \mathcal{T}$, let $\mathcal{S}(t)$ denote the set of sensitive trees $T \in G_f$ whose adjacency structure is given by t.

For conciseness let us write $s_{\text{tot}}(f)$ to denote $\sum_{x \in \{0,1\}^n} s(f,x)$. Let $\mathcal{D}$ denote the distribution on $\{0,1\}^n$ where

$$\Pr_{\mathcal{D}}[x] = \frac{s(f,x)}{s_{\text{tot}}(f)}.$$

Note that $\mathcal{D}$ is supported only on vertices where $s(f,x) \geq 1$. Further $\mathcal{D}$ is a stationary distribution for the simple random walk on G_f: if we sample a vertex from $\mathcal{D}$ and then walk to a random neighbor, it is also distributed according to $\mathcal{D}$.

Fix a tree $t \in \mathcal{T}$ and consider a random walk on G_f which is the following vector $\mathbf{X} = (\mathbf{X}_0, \ldots, \mathbf{X}_j)$ of random variables:

- We sample $\mathbf{X}_0$ from $\{0,1\}^n$ according to $\mathcal{D}$.
- For $i \geq 1$, let $\mathbf{X}_i$ be a a random neighbor of $\mathbf{X}_{i'}$ in G_f where $i' = p_t(i) < i$.

Note that every $\mathbf{X}_i$ is distributed according to $\mathcal{D}$. The vector $\mathbf{X} = (\mathbf{X}_0, \ldots, \mathbf{X}_j)$ is such that $(\mathbf{X}_i, \mathbf{X}_{p_t(i)}) \in E(G_f)$, but it might contain repeated vertices and edge labels (indeed, this proof bounds the number of homomorphisms from G_f to t).

A vector $x = (x_0, \ldots, x_j) \in (\{0,1\}^n)^{j+1}$ will be sampled with probability

$$\begin{aligned}
\Pr[\mathbf{X} = x] &= \Pr[\mathbf{X}_0 = x] \prod_{i=1}^{j} \Pr[\mathbf{X}_i = x_i | \mathbf{X}_0, \ldots, \mathbf{X}_{i-1}] \\
&= \frac{s(f,x_0)}{\sum_{x \in \{0,1\}^n} s(f,x)} \prod_{i=0}^{j-1} \frac{1}{s(f,x_i)} \\
&= \frac{1}{\sum_{x \in \{0,1\}^n} s(f,x)} \prod_{i=1}^{j-1} \frac{1}{s(f,x_i)}.
\end{aligned}$$

Clearly $\mathcal{S}(t)$ lies in the support of $\mathbf{X}$, hence

$$\begin{aligned}
|\mathcal{S}(t)| &\leq \text{supp}(\mathbf{X}) \\
&\leq \mathop{\mathbb{E}}_{\mathbf{X}} \left[\frac{1}{\Pr[\mathbf{X} = x]} \right] \\
&\leq \mathop{\mathbb{E}}_{\mathbf{X}} \left[\sum_{x \in \{0,1\}^n} s(f,x) \prod_{i=1}^{j-1} s(f, \mathbf{X}_i) \right] \\
&= s_{\text{tot}}(f) \mathop{\mathbb{E}}_{\mathbf{X}} \left[\prod_{i=1}^{j-1} s(f, \mathbf{X}_i) \right] \\
&\leq s_{\text{tot}}(f) \mathop{\mathbb{E}}_{\mathbf{Y} \sim \mathcal{D}} \left[s(f,\mathbf{Y})^{j-1} \right] \qquad (8)
\end{aligned}$$

where the last inequality holds since the $\mathbf{X}_i$'s are identically distributed and each $s(f, \mathbf{X}_i)$ is non-negative (or can be derived via the AM-GM inequality). We bound the moment under $\mathcal{D}$ as follows:

$$\begin{aligned}\mathop{\mathbb{E}}_{\mathbf{Y}\sim\mathcal{D}}\left[s(f,\mathbf{Y})^{j-1}\right] &\leq \sum_{y\in\{0,1\}^n} \Pr[\mathbf{Y}=y]s(f,y)^{j-1} \\ &= \sum_{y\in\{0,1\}^n} \frac{s(f,y)}{s_{\text{tot}}(f)}s(f,y)^{j-1} \\ &= \frac{\sum_{y\in\{0,1\}^n} s(f,y)^j}{s_{\text{tot}}(f)}.\end{aligned}$$

Plugging this back into Equation (8) gives

$$|\mathcal{S}(t)| \leq \sum_{y\in\{0,1\}^n} s(f,y)^j$$

Summing over all possibilities for t, we get

$$|\mathcal{S}_j| \leq \sum_{t\in\mathcal{T}} |\mathcal{S}(t)| \leq j! \sum_{y\in\{0,1\}^n} s(f,y)^j.$$

One can save a factor of $(j+1)$, since there are $j+1$ ways to root each tree in S_j. ◀

Theorem 4.11 now follows from an argument similar to Theorem 3.1.

Proof of Theorem 4.11. The lower bound follows from (the lower bound in) Theorem 3.1 and the observation that $\text{ts}(f_\rho) \geq s(f_\rho)$. We now prove the upper bound.

Similar to Theorem 3.1, consider the bipartite graph where the LHS is the set $\mathcal{S}_j$ of all sensitive trees T of size j in G_f, the RHS is the set $\mathcal{R}_{k,n}$ of all restrictions ρ, and (T,ρ) is an edge if the tree T lies in the subcube $C(\rho)$ specified by the restriction ρ. The desired probability $\Pr_{\boldsymbol{\rho}\in\mathcal{R}_{k,n}}[\text{ts}(f_{\boldsymbol{\rho}}) \geq j]$ is the fraction of nodes in $\mathcal{R}_{k,n}$ that are incident to at least one edge.

We first bound the degree of each vertex on the left. To have T lying in $C(\rho)$,

- The edge labels of T must be live variables for ρ.
- The values ρ_i for the fixed coordinates $i \in [n] \setminus L(\rho)$ must be consistent with the values in $V(T)$.

The only choice is of the $(k-j)$ remaining live coordinates. Hence $T \in C(\rho)$ for at most $\binom{n-j}{k-j}$ values of ρ corresponding to choices of the remaining live variables.

The number of vertices in $\mathcal{S}_j$ is bounded using Lemma 4.12 by $|\mathcal{S}_j| \leq j! \sum_{x\in\{0,1\}^n} s(f,x)^j = j!2^n s^j(f)$, so the total number of edges is at most $\binom{n-j}{k-j}2^n j! s^j(f)$. A restriction $\rho \in \mathcal{R}_{k,n}$ is specified by a set $L(\rho)$ of k live co-ordinates where $\rho_i = \star$, and a value $\rho_i \in \{0,1\}$ for the other coordinates, and hence $|\mathcal{R}_{k,n}| = \binom{n}{k}2^{n-k}$. Recall that $\text{ts}(f_\rho) \geq j$ iff $C(\rho)$ contains some tree from $\mathcal{S}_j$. Hence the fraction of restrictions ρ that have an edge incident to them is

$$\Pr_{\rho\in\mathcal{R}_{k,n}}[\text{ts}(f_\rho) \geq j] \leq \frac{\binom{n-j}{k-j}2^n j! s^j(f)}{\binom{n}{k}2^{n-k}} \leq \frac{k^j 2^k s^j(f)}{\binom{n}{j}}.$$ ◀

4.3 Applications

By combining Theorems 4.9 and 4.11, we get upper and lower bounds on the probability that a random restriction of a function has large decision tree depth in terms of its sensitivity moments.

▶ **Corollary 4.13.** *Let $f : \{0,1\}^n \to \{\pm 1\}$, $\boldsymbol{\rho} \sim \mathcal{R}_{k,n}$ and $1 \le j \le k$. Then*

$$\frac{s^{\underline{j}}(f)}{n^j} \approx \frac{s^{\underline{j}}(f)}{\prod_{i=0}^{j-1}(n-i)} \le \Pr_{\boldsymbol{\rho}\in\mathcal{R}_{k,n}}[\mathrm{dt}(f_{\boldsymbol{\rho}}) \ge j] \le \frac{(2k)^{2k} s^{\sqrt{2j}}(f)}{\prod_{i=0}^{\sqrt{2j}-2}(n-i)} \approx \frac{(2k)^{2k} s^{\sqrt{2j}}(f)}{n^{\sqrt{2j}-1}}.$$

Note that the denominator in the lower bound is $n^{\Omega(j)}$ but for the upper bound, it is $n^{\Omega(\sqrt{j})}$. This quadratic gap comes from Theorem 4.9. However, if Conjecture 4.10 stating that $\mathrm{ts}(f) \ge \mathrm{dt}(f)$ were true, it would imply the following sharper upper bound.

▶ **Corollary 4.14.** *Let $f : \{0,1\}^n \to \{\pm 1\}$, $\rho \sim \mathcal{R}_{k,n}$ and $1 \le j \le k$. If Conjecture 4.10 holds, then*

$$\Pr_{\boldsymbol{\rho}\in\mathcal{R}_{k,n}}[\mathrm{dt}(f_{\boldsymbol{\rho}}) \ge j] \le \frac{(2k)^{2k} s^{j}(f)}{\prod_{i=0}^{j-1}(n-i)} \approx \frac{(2k)^{2k} s^{j}(f)}{n^j}.$$

The dependence on n here matches that in the lower bound of Corollary 4.13. Conjecture 1.3 follows from this as an easy consequence (indeed showing $\mathrm{ts}(f) \ge \deg(f)$ rather than Conjecture 4.10 suffices):

▶ **Corollary 4.15.** *Conjecture 4.10 implies Conjecture 1.3.*

Proof. We will prove that $\mathbb{I}^{\underline{k}}(f) \le (2k)^{2k} s^k(f)$. Let $\boldsymbol{\rho} \leftarrow \mathcal{R}_{k,n}$ and consider the event that $\deg(f_{\boldsymbol{\rho}}) = k$. By Theorem 3.2, we can lower bound this probability in terms of the Fourier moments of f as

$$\frac{\mathbb{I}^{\underline{k}}(f)}{\prod_{i=0}^{k-1}(n-i)} \le \Pr_{\boldsymbol{\rho}\leftarrow\mathcal{R}_{k,n}}[\deg(f_{\boldsymbol{\rho}}) = k].$$

To upper bound it, by Corollary 4.14, if Conjecture 4.10 holds, then we have

$$\Pr_{\boldsymbol{\rho}\in\mathcal{R}_{k,n}}[\deg(f_{\boldsymbol{\rho}}) \ge k] \le \Pr_{\boldsymbol{\rho}\in\mathcal{R}_{k,n}}[\mathrm{dt}(f_{\boldsymbol{\rho}}) \ge k] \le \frac{(2k)^{2k} s^{k}(f)}{\prod_{i=0}^{k-1}(n-i)}.$$

The claim follows by comparing the upper and lower bounds. ◀

For $k = 3, 4$, it is an easy exercise to verify that $\mathrm{dt}(f_\rho) = k$ implies $\mathrm{ts}(f_\rho) = k$. This implies that Conjecture 1.3 holds for $k = 3, 4$.

We conclude this section with an application to the class of width-w DNF formulas. In Section 3 we showed how the switching lemma implies sensitivity moment bounds for DNFs (and AC^0). Here we show the converse, how a version of the switching lemma can be derived using sensitivity moment bounds. The Satisfiability Coding Lemma of [21] implies the following moment bounds for DNFs:

▶ **Lemma 4.16.** *[21] There exists a constant c such that if f has a width-w DNF formula, then $s^k(f) \le (ckw)^k$.*

([21] proved tail bounds on the sensitivity of small-width DNFs from which a simple calculation leads to the above moment bound. We refer the reader to [12] for more details, and for an example showing the tightness of this bound.)

If Conjecture 4.10 holds, plugging these bounds into Corollary 4.14 gives that for any width-w DNF f, there exists $c', c'' > 0$ such that

$$\Pr_{\boldsymbol{\rho}\in\mathcal{R}_{k,n}}[\mathrm{dt}(f_{\boldsymbol{\rho}}) \ge k] \le \frac{(c'k)^{3k} w^k}{\prod_{i=0}^{k-1}(n-i)} \approx \left(\frac{c''k^3 w}{n}\right)^k.$$

This nearly matches the bound one gets from Håstad's switching lemma (with k^3 in place of k). Thus proving Conjecture 4.10 would give a combinatorial proof of the switching lemma for DNFs which seems very different from the known proofs of Håstad [14] and Razborov [23].

5 Proper Walks

Since $s^j(f) \leq (s(f))^j$ for all j, one can trivially bound the sensitivity moments of a function in terms of its max sensitivity. Hence Corollaries 4.14 and 4.15 show that under Conjecture 4.10, low sensitivity functions simplify under random restrictions. In this section we prove this unconditionally. The key ingredient is a relaxation of sensitive trees that we call *proper walks*.

A *walk* W in the n-dimensional Boolean cube is a sequence of vertices $(w_0, w_1, \ldots, w_t)$ such that w_i and w_{i+1} are at Hamming distance precisely 1. We allow walk to backtrack and visit vertices more than once. We say that t is the *length* of such a walk.

Let $\ell(W) \subseteq [n]$ denote the set of coordinates that are flipped by walk W. We define $k = |\ell(W)|$ to be the *dimension* of the walk. We order the coordinates in $\ell(W)$ as $\ell_1, \ldots, \ell_k$ according to the order in which they are first flipped. For each $\ell_i \in \ell(W)$, let x_i denote the first vertex in W at which we flip coordinate i.

▶ **Definition 5.1.** A walk W is a *proper walk* for a Boolean function $f : \{0,1\}^n \to \{\pm 1\}$ if for each $\ell_i \in \ell(W)$, the vertex x_i is sensitive to ℓ_i.

Thus a walk is proper for f if the first edge flipped along a new coordinate direction is always sensitive. This implies that while walking from x_i to x_{i+1}, we are only allowed to flip a subset of the coordinates $\{\ell_1, \ldots, \ell_i\}$, hence $\mathrm{supp}(x_i \oplus x_{i+1}) \subseteq \{\ell_1, \ldots, \ell_i\}$. Hence if there is a proper walk of dimension k then there is one of length at most $k(k+1)/2$, by choosing a shortest path between x_i and x_{i+1} for each i.

In studying proper walks, it is natural to try to maximize the dimension and minimize the length. We first focus on the former. The following lemma states that the obvious necessary condition for the existence of an n-dimensional walk is in fact also sufficient:

▶ **Lemma 5.2.** *Every Boolean function $f : \{0,1\}^n \to \{\pm 1\}$ that depends on all n coordinates has a proper walk of dimension n.*

Proof. Pick $\ell_1 \in [n]$ arbitrarily and let x_1 be any vertex in $\{0,1\}^n$ which is sensitive to coordinate ℓ_1. Let $1 \leq i \leq n$. Inductively we assume we have picked coordinates $L = \{\ell_1, \ldots, \ell_i\}$ and points $X = \{x_1, \ldots, x_i\}$ so that for every $j \leq i$,

1. x_j is sensitive to ℓ_j.
2. For $j \geq 2$, $\mathrm{supp}(x_{j-1} \oplus x_j) \subseteq \{\ell_1, \ldots, \ell_{j-1}\}$.

If we visit $x_1, \ldots, x_i$ in that order and walk from each x_j to x_{j+1} along a shortest path, the resulting walk is a proper walk for f. Let C be the subcube that spans the dimensions in L and contains X.

Case 1: Some vertex in C is sensitive to a coordinate outside of L. Name this vertex x_{i+1} and the sensitive co-ordinate ℓ_{i+1}, and add them to X and L repectively. Note that $x_i \oplus x_{i+1}$ is indeed supported on $\{\ell_1, \ldots, \ell_i\}$, so both conditions (1) and (2) are met.

Case 2: No vertex in C is sensitive to a coordinate outside L. So for any co-ordinate $j \notin L$, we have $f(x) = f(x \oplus e_j)$. But this means that the set of points $X \oplus e_j$ and co-ordinates L also satify the inductive hypothesis (specifically conditions (1) and (2) above).

Let d denote the Hamming distance from C to the closest vertex which is sensitive to some coordinate outside L. Let z denote one such closest vertex to C (there could be many) and pick any coordinate j in which z differs from the closest point in C. If we replace X by $X \oplus e_j$, the Hamming distance to z has decreased to $d-1$. We can repeat this till the Hamming distance drops to 0, which puts us in Case (1). ◀

Given this result, it is natural to try to find full dimensional walks of the smallest possible length. The length of the walk constructed above is bounded by $\sum_{i=1}^{n}(i-1) \leq n^2/2$. Lemma A.2 in Appendix A gives an example showing that this is tight up to constants. So while we cannot improve the bound in general, we are interested in the case of functions with large decision tree complexity, where the following observation suggests that better bounds should be possible.

▶ **Lemma 5.3.** *If* $\mathrm{ts}(f) = n$*, then* f *has a proper walk of dimension* n *and length* $2n-1$*.*

The proof is by doing a pre-order traversal of a sensitive tree of dimension n. Thus if Conjecture 4.10 were true, it would imply that functions requiring full decision tree depth have proper walks of length $O(n)$. We now give an unconditional proof of this result (we will use it as an essential ingredient in our "switching lemma" later).

▶ **Theorem 5.4.** *If* $\mathrm{dt}(f) = n$*, then* f *has a proper walk of dimension* n *and length at most* $3n$*.*

Proof. The proof is by induction on n. The base case $n = 2$ is trivial since in this case there exists a proper walk of length 2. Assume the claim holds for all $n' < n$. Let f be a function where $\mathrm{dt}(f) = n$. If $\mathrm{ts}(f) = n$ we are done by Lemma 5.3, so we assume that $\mathrm{ts}(f) = m < n$. By Corollary 4.7, there is an orchard $\{T \oplus v\}$ of sensitive trees where $\dim(T) = m$. Assume by relabeling that $\ell(T) = \{1, \ldots, m\}$.

Since $\mathrm{dt}(f) = n$, there exists a setting $t_1, \ldots, t_m$ of variables in $[m]$ such that the restriction $f' = f|_{x_1=t_1,\ldots,x_m=t_m}$ on $n' = n-m$ variables satisfies $\mathrm{dt}(f') = n-m$. By the inductive hypothesis, there exists a proper walk in f' of dimension $n-m$ and length $3(n-m)$ in the subcube $x_1 = t_1, \ldots, x_m = t_m$ which starts at some vertex $s' = (t_1, \ldots, t_m, s'_{m+1}, \ldots, s'_n)$ and ends at some vertex $t' = (t_1, \ldots, t_m, t'_{m+1}, \ldots, t'_n)$, which flips all coordinates in $[n] \setminus [m]$.

Consider the tree $T \oplus v$ in the orchard such that the coordinates of $V(T \oplus v)$ in $[n] \setminus [m]$ agree with s'. Our walk can be divided into three phases:

1. By Lemma 5.3, we can visit every vertex in $T \oplus v$ using a proper walk of length at most $2m-1$ that only uses edges in $[m]$. Assume that this walk starts at a and ends at b. By our choice of v we have that $(b_{m+1}, \ldots, b_n) = (s'_{m+1}, \ldots, s'_n)$.
2. From b, we then walk to the vertex $s = (t_1, \ldots, t_m, s'_{m+1}, \ldots, s'_n)$. This only requires flipping bits in $[m]$, so it keeps the walk proper and adds only m to its length.
3. The inductive hypothesis applied to f' allows us to construct a proper walk from s to t that only walks along edges in $[n] \setminus [m]$ and has length at most $3(n-m)$.

Thus the total length of the walk is bounded by $2m - 1 + m + 3(n-m) < 3n$. ◀

5.1 Random Restrictions of Low Sensitivity Functions

In this section we prove our "switching lemma for low-sensitivity functions," Lemma 5.6. The high-level idea is to study the existence of (short) proper walks for a random restriction $f_{\boldsymbol{\rho}}$ of f, and use Theorem 5.4 to transfer a bound on the probability that f_ρ has such short proper walks, to a bound on the probability that f_ρ has full decision tree depth. Similar in

spirit to Theorem 4.11, the proof proceeds by grouping walks according to their *topology*, and showing that $f_{\boldsymbol{\rho}}$ is unlikely to contain any of them. We now define the notion of a "walk topology":

▶ **Definition 5.5.** A *walk topology* of dimension k and length ℓ is a sequence $\mathrm{wt} = (\mathrm{wt}_1, \ldots, \mathrm{wt}_\ell)$ of coordinates in $[k]$ (possibly with repetitions), where all elements of $[k]$ appear in wt, and they first appear in the order $1, \ldots, k$.

Given a walk topology wt, a starting point $x_0 \in \{0,1\}^n$, and a sequence $L = (\ell_1, \ldots, \ell_k)$ of distinct coordinates in $[n]$, we get a walk $W = W(x_0, L, \mathrm{wt})$ on the n-dimensional cube by starting at x_0 and associating label i of w with coordinate ℓ_i for $i \in [k]$. The walk W has length ℓ and dimension k. Conversely, every walk W gives a unique triple (x_0, L, wt) corresponding to its starting point, order in which coordinates are first flipped, and its topology.

▶ **Lemma 5.6.** *Let $f : \{0,1\}^n \to \{\pm 1\}$. Then*

$$\Pr_{\boldsymbol{\rho} \in \mathcal{R}_{k,n}}[\mathrm{dt}(f_{\boldsymbol{\rho}}) = k] \leq \frac{(2k^3 s(f))^k}{\prod_{i=0}^{k-1}(n-i)}.$$

Proof. Fix a restriction $\rho \in \mathcal{R}_{k,n}$. Theorem 5.4 implies that if $\mathrm{dt}(f_\rho) = k$, then f_ρ contains a proper walk W of dimension k and length $3k$. Let $\mathcal{TOP}$ denote the set of all walk topologies of dimension k and length $3k$, so that $|\mathcal{TOP}| \leq k^{3k}/k!$. (Observe that of the k^{3k} strings in $[k]^{3k}$, precisely a $1/k!$ fraction of those in which all k elements appear will have them appearing first in the order $1, \ldots, k$.) Let $\mathbb{S}_K$ denote the set of permutations of the live variables K of ρ.

Fix $\mathrm{wt} \in \mathcal{TOP}$. We say that wt is *good for* $\rho \in \mathcal{R}_{k,n}$ if there exists a proper walk for f_ρ with topology wt; in other words there exists $y \in C(\rho)$ and $L \in \mathbb{S}_K$ such that $W = W(y, L, \mathrm{wt})$ is a proper walk for f. To bound the probability of this event, we first show that it suffices to consider the case when y and L are uniformly random: we have that $\Pr_{\boldsymbol{\rho} \in \mathcal{R}_{k,n}}[\mathrm{wt} \text{ is good for } f_{\boldsymbol{\rho}}]$ equals

$$\begin{aligned}&\Pr_{\boldsymbol{\rho} \in \mathcal{R}_{k,n}}[\exists y \in C(\boldsymbol{\rho}), L \in \mathbb{S}_K \text{ s.t. } W = W(y, L, \mathrm{wt}) \text{ is a proper walk for } f_{\boldsymbol{\rho}}] \\ &\leq k! 2^k \Pr_{\boldsymbol{\rho} \leftarrow \mathcal{R}_{k,n}, \boldsymbol{y} \leftarrow C(\boldsymbol{\rho}), \mathbf{L} \leftarrow \mathbb{S}_K}[W(\boldsymbol{y}, \mathbf{L}, \mathrm{wt}) \text{ is a proper walk for } f_{\boldsymbol{\rho}}], \end{aligned} \tag{9}$$

where the inequality holds since for each outcome ρ of $\boldsymbol{\rho}$ there are 2^k points $y \in C(\rho)$ and $k!$ elements $L \in \mathbb{S}_K$.

Sampling the triple $(\boldsymbol{\rho}, \boldsymbol{y}, \mathbf{L})$ is equivalent to independently sampling $\boldsymbol{x} \leftarrow \{0,1\}^n$ and $\mathbf{L} = (\boldsymbol{\ell}_1, \ldots, \boldsymbol{\ell}_k)$ by picking k coordinates uniformly from $[n]$ without replacement. It is easy to see that this determines $(\boldsymbol{\rho}, \boldsymbol{y}, \mathbf{L})$ and hence the walk $\mathbf{W} = W(\boldsymbol{y}, \mathbf{L}, \mathrm{wt})$. We can now define the sequence of points on the walk $\mathbf{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k)$ as described earlier so that $\mathbf{W}$ is proper for $f_{\boldsymbol{\rho}}$ if $\boldsymbol{x}_i$ is sensitive to $\boldsymbol{\ell}_i$. Hence

$$\begin{aligned}\Pr_{\boldsymbol{x}, \mathbf{L}}[\mathbf{W} \text{ is a proper walk for } f_{\boldsymbol{\rho}}] &= \Pr_{\boldsymbol{x}, \mathbf{L}}[\boldsymbol{x}_i \text{ is sensitive to } \boldsymbol{\ell}_i \ \forall i \in [k]] \\ &= \prod_{i \leq k} \Pr_{\boldsymbol{x}, \mathbf{L}}[\boldsymbol{x}_i \text{ is sensitive to } \boldsymbol{\ell}_i \big| \boldsymbol{x}_j \text{ is sensitive to } \boldsymbol{\ell}_j \ \forall j < i].\end{aligned} \tag{10}$$

Let us first sample $\boldsymbol{x} \leftarrow \{0,1\}^n$, and then sample the elements of $\mathbf{L} = (\boldsymbol{\ell}_1, \ldots, \boldsymbol{\ell}_k)$ one at a time without replacement. Observe that $\boldsymbol{x}_i$ (the first time on the walk $\mathbf{W}$ that we flip

$\boldsymbol{\ell}_i$) is a function of $\boldsymbol{x}, \boldsymbol{\ell}_1, \ldots, \boldsymbol{\ell}_{i-1}$, since $\boldsymbol{x}_1 = \boldsymbol{x}$ and $\mathrm{supp}(\boldsymbol{x}_i \oplus \boldsymbol{x}_1) \subseteq \{\boldsymbol{\ell}_1, \ldots, \boldsymbol{\ell}_{i-1}\}$ (and the exact subset is specified by wt). Hence fixing outcomes $x, \ell_1, \ldots, \ell_{i-1}$ of $\boldsymbol{x}, \boldsymbol{\ell}_1, \ldots, \boldsymbol{\ell}_{i-1}$ fixes the random variables $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_i$ and whether $\boldsymbol{x}_j$ is sensitive to $\boldsymbol{\ell}_j$ for $j < i$ (the events that we condition on in Equation 10). We then sample $\boldsymbol{\ell}_i$ uniformly from the coordinates in $[n] \setminus \{\ell_1, \ldots, \ell_{i-1}\}$; crucially, $\boldsymbol{x}_i$ is sensitive to at most $s(f)$ of these coordinates. Hence

$$\Pr_{\boldsymbol{x},\mathbf{L}}[\boldsymbol{x}_i \text{ is sensitive to } \boldsymbol{\ell}_i | \boldsymbol{x}_j \text{ is sensitive to } \ell_j \ \forall j < i] \leq \frac{s(f)}{n-i+1}.$$

Plugging this into Equation (10),

$$\Pr_{\boldsymbol{x},\mathbf{L}}[\mathbf{W} \text{is a proper walk for } f_{\boldsymbol{\rho}}] \leq \frac{(s(f))^k}{\prod_{i=0}^{k-1}(n-i)}. \tag{11}$$

Hence by Equation (9),

$$\Pr_{\boldsymbol{\rho}\in\mathcal{R}_{k,n}}[\mathrm{wt} \text{ is good for } f_{\boldsymbol{\rho}}] \leq \frac{k! 2^k (s(f))^k}{\prod_{i=0}^{k-1}(n-i)}. \tag{12}$$

Taking a union bound over all (at most) $k^{3k}/k!$ possible choices of $\mathrm{wt} \in \mathcal{TOP}$, we get that

$$\Pr_{\boldsymbol{\rho}\in\mathcal{R}_{k,n}}[\mathrm{dt}(f_\rho) = k] \leq \frac{k^{3k} 2^k (s(f))^k}{\prod_{i=0}^{k-1}(n-i)} \leq \frac{(2k^3 s(f))^k}{\prod_{i=0}^{k-1}(n-i)}. \quad \blacktriangleleft$$

We note that one can prove a similar bound for $\Pr_{\rho\in\mathcal{R}_{k,n}}[\mathrm{dt}(f_\rho) \geq j]$; here we have presented only the case $j = k$ both because it is simpler and because it suffices for the concentration results in Section 5.2.

We would like to replace the $(s(f))^k$ term with $s^k(f)$, the k^{th} sensitivity moment. The above proof does not seem to generalize to that case, because we do not have an analogue of Sidorenko's result on trees for proper walks.

5.2 Fourier tails of low sensitivity functions

We have the necessary pieces in place to give an upper bound on $\mathbb{I}^{\mathrm{k}}[f]$:

▶ **Lemma 5.7.** *For every $f : \{\pm1\}^n \to \{\pm1\}$ and every $k \geq 1$, we have $\mathbb{I}^{\mathrm{k}}[f] \leq (2k^3 s(f))^k$.*

Proof. By Theorem 3.2 and Lemma 5.6, we have that

$$\frac{\mathbb{I}^{\mathrm{k}}[f]}{\prod_{i=0}^{k-1}(n-i)} \leq \Pr_{\rho\in\mathcal{R}_{k,n}}[\deg(f_\rho) = k] \leq \Pr_{\rho\in\mathcal{R}_{k,n}}[\mathrm{dt}(f_\rho) = k] \leq \frac{(2k^3 s(f))^k}{\prod_{i=0}^{k-1}(n-i)},$$

which may be rewritten as the claimed bound. ◀

Next we observe that bounding $\mathbb{I}^{\mathrm{k}}[f]$ yields tail bounds for the Fourier spectrum of f.

▶ **Lemma 5.8.** *For every $f : \{\pm1\}^n \to \{\pm1\}$, every $k \geq 1$, and every $\epsilon > 0$, we have*

$$\deg_\epsilon(f) \leq \max\left(k, e\left(\frac{\mathbb{I}^{\mathrm{k}}[f]}{\epsilon}\right)^{1/k}\right).$$

Proof. We first consider the case when $\mathbb{I}^{\mathrm{k}}[f]/k! \leq \epsilon$. In this case,

$$\sum_{|S|\geq k} \hat{f}(S)^2 \leq \sum_{|S|\geq k} \hat{f}(S)^2 \binom{|S|}{k} = \frac{\mathbb{I}^{\mathrm{k}}[f]}{k!} \leq \epsilon$$

so $\deg_\epsilon(f) \leq k$.

So assume that $\mathbb{I}^{\underline{k}}[f]/k! \geq \epsilon$. It suffices to prove that for

$$t_0 = e\left(\frac{\mathbb{I}^{\underline{k}}[f]}{\epsilon}\right)^{1/k}$$

we have

$$\sum_{|S|\geq t_0} \hat{f}(S)^2 = \Pr_{\mathbf{R}\leftarrow\mathcal{D}_f}[|\mathbf{R}| \geq t_0] \leq \epsilon.$$

Since $\binom{t}{k}$ is strictly incresing for $t \geq k$, we have

$$\Pr_{\mathbf{R}\leftarrow\mathcal{D}_f}[|\mathbf{R}| \geq t] = \Pr_{\mathbf{R}\leftarrow\mathcal{D}_f}\left[\binom{|\mathbf{R}|}{k} \geq \binom{t}{k}\right].$$

Now observe that we have

$$\frac{\mathbb{I}^{\underline{k}}[f]}{k!} = \mathop{\mathbb{E}}_{\mathbf{R}\leftarrow\mathcal{D}_f}\left[\binom{|\mathbf{R}|}{k}\right].$$

Hence for any t such that

$$t \geq k\left(\frac{\mathbb{I}^{\underline{k}}[f]}{k!\epsilon}\right)^{1/k} \geq k$$

Markov's inequality gives

$$\Pr_{\mathbf{R}\leftarrow\mathcal{D}_f}[|\mathbf{R}| \geq t] \leq \frac{\mathbb{E}_{\mathbf{R}\leftarrow\mathcal{D}_f}[\binom{|\mathbf{R}|}{k}]}{\binom{t}{k}} \leq \frac{\mathbb{I}^{\underline{k}}[f]}{k!\cdot(t/k)^k} \leq \epsilon,$$

One can check that t_0 satisfies the required bound using Stirling's approximation. ◀

Now we are ready to prove Theorem 1.2:

▶ **Theorem 1.2** (restated). *For any function f and any $\epsilon > 0$, we have* $\deg_\epsilon(f) \leq O(s(f)(\log 1/\epsilon)^3)$.

Proof. Applying Lemma 5.8 and Lemma 5.7, for every $f : \{\pm1\}^n \to \{\pm1\}$, every $k \geq 1$, and every $\epsilon > 0$, we have

$$\deg_\epsilon(f) \leq \max\left\{k, e\frac{2k^3 s(f)}{\epsilon^{1/k}}\right\} \leq 2es(f)\frac{k^3}{\epsilon^{1/k}}.$$

Taking $k = \log(1/\epsilon)$, we get that

$$\deg_\epsilon(f) = O(s(f)\log(1/\epsilon)^3),$$

as claimed. ◀

We note that the relations between influence moments and Fourier concentration that are established in [26, Section 4] can also be used to obtain Theorem 1.2 from Lemma 5.7. [26, Section 4] also shows that bounded k-th influence moments imply bounded Fourier L_1 spectral norm on the k-th level, which in turn implies Fourier concentration on a small number of Fourier coefficients (smaller than the trivial $\binom{n}{k}$ bound on the number of coefficients up to degree k). These results can be used with Lemma 5.7 to establish the corresponding Fourier bounds for functions with bounded max sensitivity.

6 Additional questions and complexity measures

As stated earlier, we hope that this work will stimulate further research on the sensitivity graph G_f and on complexity measures associated with it. Towards this end we conclude with some additional questions and a new complexity measure.

The graph G_f consists of a number of connected components. This component structure naturally suggests another complexity measure:

▶ **Definition 6.1.** For $x \in \{0,1\}^n$, the *component dimension of f at x*, denoted $\mathrm{cdim}(f,x)$, is the dimension of the connected component of G_f that contains x (i.e. the number of coordinates i such that x's component contains at least one edge in the i-th direction). We define $\mathrm{cdim}(f)$ to be $\max_{x\in\{0,1\}^n} \mathrm{cdim}(f,x)$).

It is easy to see that $\mathrm{cdim}(f) \geq \mathrm{ts}(f) \geq s(f)$, and thus a consequence of Conjecture 4.10 is that $\mathrm{cdim}(f) \geq \mathrm{dt}(f)$; however we have not been able to prove a better lower bound for $\mathrm{cdim}(f)$ in terms of $\mathrm{dt}(f)$ than that implied by Theorem 4.9. We note that $\mathrm{cdim}(f)$ and $\mathrm{ts}(f)$ are not polynomially related, since the addressing function shows that the gap between them can be exponential.

Lastly, it is an intriguing open question whether the reverse direction of the robust sensitivity conjecture also holds: for every k, does there exist a'_k, b'_k such that $\mathbb{E}[\boldsymbol{s}^k] \leq a'_k\,\mathbb{E}[\boldsymbol{d}^k] + b'_k$? Can one relate this question to a statement about graph-theoretic (or other) complexity measures?

Acknowledgments. We thank Yuval Peres and Laci Lovasz for pointing us to Sidorenko's theorem and the related literature. We also thank Yuval for useful discussions and for letting us present his proof of Lemma 4.12 here. We thank Gagan Aggarwal for showing us a combinatorial proof of Sidorenko's theorem. We thank Avishay Tal for helpful comments and suggestions.

References

1 Andris Ambainis, Mohammad Bavarian, Yihan Gao, Jieming Mao, Xiaoming Sun, and Song Zuo. Tighter relations between sensitivity and other complexity measures. In *Automata, Languages, and Programming – 41st International Colloquium, ICALP 2014*, pages 101–113, 2014.

2 Andris Ambainis and Krisjanis Prusis. A tight lower bound on certificate complexity in terms of block sensitivity and sensitivity. In *MFCS*, pages 33–44, 2014.

3 Andris Ambainis, Krisjanis Prusis, and Jevgenijs Vihrovs. Sensitivity versus certificate complexity of boolean functions. *CoRR*, abs/1503.07691, 2015.

4 Andris Ambainis and Jevgenijs Vihrovs. Size of Sets with Small Sensitivity: a Generalization of Simon's Lemma. In *Theory and Applications of Models of Computation – 12th Annual Conference, TAMC 2015*, pages 122–133, 2015.

5 M. Blum and R. Impagliazzo. Generic oracles and oracle classes. In *Proceedings of the 28th Annual Symposium on Foundations of Computer Science*, pages 118–126, 1987.

6 H. Buhrman and R. de Wolf. Complexity measures and decision tree complexity: a survey. *Theoretical Computer Science*, 288(1):21–43, 2002.

7 S. Chakraborty, R. Kulkarni, S. Lokam, and N. Saurabh. Upper Bounds on Fourier Entropy. ECCC report TR13-052 Revision #1, available at http://eccc.hpi-web.de/report/2013/052/, 2013.

8 P. Csikvári and Z. Lin. Graph homomorphisms between trees. *Electronic Journal of Combinatorics*, 21(4):P4.9, 2014.

9 E. Friedgut. Boolean functions with low average sensitivity depend on few coordinates. *Combinatorica*, 18(1):474–483, 1998.

10 Ehud Friedgut and Gil Kalai. Every monotone graph property has a sharp threshold. *Proceedings of the American mathematical Society*, 124(10):2993–3002, 1996.

11 P. Gopalan, N. Nisan, R. Servedio, K. Talwar, and A. Wigderson. Smooth boolean functions are easy: efficient algorithms for low sensitivity functions. In *ITCS*, pages 59–70, 2016.

12 P. Gopalan, R. Servedio, A. Tal, and A. Wigderson. Degree and Sensitivity: tails of two distributions. to appear, 2016.

13 J. Hartmanis and L.A. Hemachandra. One-way functions, robustness and non-isomorphism of NP-complete classes. *Theor. Comput. Sci*, 81(1):155–163, 1991.

14 J. Håstad. *Computational Limitations for Small Depth Circuits*. MIT Press, Cambridge, MA, 1986.

15 Pooya Hatami, Raghav Kulkarni, and Denis Pankratov. *Variations on the Sensitivity Conjecture*. Number 4 in Graduate Surveys. Theory of Computing Library, 2011. URL: `http://www.theoryofcomputing.org/library.html`, `doi:10.4086/toc.gs.2011.004`.

16 J. Kahn, G. Kalai, and N. Linial. The influence of variables on boolean functions. In *Proc. 29th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 68–80, 1988.

17 Claire Kenyon and Samuel Kutin. Sensitivity, block sensitivity, and l-block sensitivity of Boolean functions. *Information and Computation*, pages 43–53, 2004.

18 N. Linial, Y. Mansour, and N. Nisan. Constant depth circuits, Fourier transform and learnability. *Journal of the ACM*, 40(3):607–620, 1993.

19 N. Nisan and M. Szegedy. On the degree of Boolean functions as real polynomials. *Comput. Complexity*, 4:301–313, 1994.

20 Noam Nisan. CREW PRAMs and decision trees. *SIAM Journal on Computing*, 20(6):999–1007, 1991.

21 R. Paturi, P. Pudlák, and F. Zane. Satisfiability coding lemma. In *38th Annual Symposium on Foundations of Computer Science, FOCS'97*, pages 566–574, 1997.

22 Y. Peres. Personal communication. 2015.

23 Alexander Razborov. Bounded arithmetic and lower bounds in Boolean complexity. In *Feasible Mathematics II*, pages 344–386. Springer, 1995.

24 A. Sidorenko. A partially ordered set of functionals corresponding to graphs. *Discrete Mathematics*, 131(1-3):263–277, 1994.

25 A. Tal. Shrinkage of de Morgan Formulae from Quantum Query Complexity. ECCC report TR14-048, available at `http://eccc.hpi-web.de/report/2014/048/`, 2014.

26 A. Tal. Tight Bounds on The Fourier Spectrum of AC^0. ECCC report TR14-174 Revision #1, available at http://eccc.hpi-web.de/report/2014/174/, 2015.

27 G. Tardos. Query complexity, or why is it difficult to separate $NP^A \cap co - NP^A$ from P^A by a random oracle A? *Combinatorica*, 8(4):385–392, 1989.

A Some Examples

▶ **Lemma A.1.** *Let $n = 2^k - 1$. There exists $f : \{0,1\}^n \to \{\pm 1\}$ for which* $\mathrm{dt}(f) = \log(n+1)$ *whereas $ts(f) = n$.*

Proof. Take a complete binary tree with n internal nodes and $n+1$ leaves. The leaves are alternately labelled 1 and -1 from left to right, while the internal nodes are labelled with $x_1, \ldots, x_n$ according to an in-order traversal of the tree. The bound on decision tree depth follows from the definition of f. To lower bound $\mathrm{ts}(f)$, we start at the -1^n input and start flipping bits from -1 to 1 in the order $x_1, \ldots, x_n$. It can be verified that every bit flip changes the value of the function. ◀

▶ **Lemma A.2.** *There exists a Boolean function f on n variables such that any proper walk for f has length $\Omega(n^2)$.*

Proof. Assume that n is a power of 2 and fix a Hadamard code of length $n/2$. We define an n-variable function f over variables $x_1, \ldots, x_{n/2}$ and $y_1, \ldots, y_{n/2}$ as follows: if the string $x_1, \ldots, x_{n/2}$ equals the i-th codeword in the Hadamard code of length $n/2$, then the output is y_i, otherwise the output is 0. Note that for any $i \neq j$, if n-bit inputs a, b are sensitive to y_i, y_j respectively then the Hamming distance between a and b must be at least $n/4$. Thus any proper walk must flip at least $n/4$ bits between any two vertices that are sensitive to different y_is, so the minimum length of any proper walk must be at least $n^2/8$. ◀

New Hardness Results for Graph and Hypergraph Colorings*

Joshua Brakensiek[1] and Venkatesan Guruswami[2]

1 **Department of Mathematical Sciences, Carnegie Mellon University, Pittsburgh, USA**
jbrakens@andrew.cmu.edu

2 **Computer Science Department, Carnegie Mellon University, Pittsburgh, USA**
guruswami@cmu.edu

Abstract

Finding a proper coloring of a t-colorable graph G with t colors is a classic NP-hard problem when $t \geq 3$. In this work, we investigate the approximate coloring problem in which the objective is to find a proper c-coloring of G where $c \geq t$. We show that for all $t \geq 3$, it is NP-hard to find a c-coloring when $c \leq 2t - 2$. In the regime where t is small, this improves, via a unified approach, the previously best known hardness result of $c \leq \max\{2t - 5, t + 2\lfloor t/3 \rfloor - 1\}$ [9, 21, 13]. For example, we show that 6-coloring a 4-colorable graph is NP-hard, improving on the NP-hardness of 5-coloring a 4-colorable graph.

We also generalize this to related problems on the strong coloring of hypergraphs. A k-uniform hypergraph H is t-strong colorable (where $t \geq k$) if there is a t-coloring of the vertices such that no two vertices in each hyperedge of H have the same color. We show that if $t = \lceil 3k/2 \rceil$, then it is NP-hard to find a 2-coloring of the vertices of H such that no hyperedge is monochromatic. We conjecture that a similar hardness holds for $t = k + 1$.

We establish the NP-hardness of these problems by reducing from the hardness of the Label Cover problem, via a "dictatorship test" gadget graph. By combinatorially classifying all possible colorings of this graph, we can infer labels to provide to the label cover problem. This approach generalizes the "weak polymorphism" framework of [3], though interestingly our results are "PCP-free" in that they do not require any approximation gap in the starting Label Cover instance.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases hardness of approximation, graph coloring, hypergraph coloring, polymorphisms, combinatorics

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.14

1 Introduction

A t-coloring of a graph $G = (V, E)$ is a coloring of its vertices with t colors such that the endpoints of every edge receive distinct colors, i.e., a map $c : V \to \{1, 2, \ldots, t\}$ such that or every $(u, v) \in E$, $c(u) \neq c(v)$. The chromatic number of a graph G, denoted $\chi(G)$, is the minimum t for which G admits a t-coloring. A graph G is said to be t-colorable if $\chi(G) \leq t$. For $t \geq 3$, finding a t-coloring of a t-colorable graph is one of the classic NP-hard problems. The problem remains difficult even when one is allowed to use many more colors. In fact, the best known efficient algorithms to color a 3-colorable graph require $n^{\Omega(1)}$ colors. However, the known NP-hardness results only rule out coloring a 3-colorable graph with a mere 4

* Research supported in part by NSF grants CCF-0963975, CCF-1115525 and CCF-1526092.

COMPUTATIONAL COMPLEXITY CONFERENCE

colors [21, 13]. By an easy reduction on this implies the NP-hardness of coloring a t-colorable graph with $t + 2\lfloor t/3 \rfloor - 1$ colors. The work of Garey and Johnson [9] gave an elegant reduction from 3-colorability using Kneser graphs to show NP-hardness of $(2t-5)$-coloring a t-colorable graph, for all $t \geq 6$.[1] Much stronger hardness results are known for larger t, and as well as conditional hardness results for $t = 3, 4$ under variants of the Unique Games conjecture; we review some of the literature on inapproximability of graph/hypergraph coloring in Section 1.1.

In this work, we prove the following NP-hardness result for coloring t-colorable graphs which improves the previous best known result in the challenging regime where t is small. The result holds for graphs whose degree is bounded by a function of t, which can be taken to be $5t^6$.

▶ **Theorem 1.1.** *For every $t \geq 3$, it is NP-hard to distinguish, given an input graph G, whether $\chi(G) \leq t$ or $\chi(G) \geq 2t-1$. In particular, $(2t-2)$-coloring a t-colorable graph is NP-hard.*

While the above does not improve the state of affairs for $t = 3$, it does yield new results for other small t, such as the NP-hardness of 6-coloring a 4-colorable graph.[2] We also note that by plugging in the NP-hardness of telling if $\chi(G) \leq 3$ or $\chi(G) \geq 5$ from [21, 13] as the starting point in the reduction of Garey and Johnson [9], together with bounds for multicoloring Kneser graphs [26], one can show that it is NP-hard to $2t-3$-color a t-colorable graph for $t \geq 6$ (improving the $2t-5$ bound in [9]).

The improvement in Theorem 1.1 is quantitatively modest, but we feel our proof methodology reveals insights into the source of the hardness, and also gives results stronger than previous works for small t in a *unified* manner. Our reduction is inspired by techniques used to show hardness of constraint satisfaction problems and employs dictatorship gadgets in a modular fashion, and the analysis hinges on combinatorial arguments to classify colorings of the gadget (more about our techniques in Section 1.2). It is worth pointing out that Theorem 1.1, as well as our results for hypergraph coloring below, are "PCP-free" in that they reduce from standard NP-hardness results for decision problems (as opposed to promise problems with an approximation gap in the optimal value). This is also true of the hardness results in [9, 13].

Hypergraph coloring. We also prove new hardness results for coloring hypergraphs. We will be interested in k-uniform hypergraphs for small k where each hyperedge has exactly k vertices. A hypergraph is *t-colorable* if its vertices can be colored with t colors so that there is no monochromatic hyperedge. We say that a hypergraph is *t-strong colorable* (or *t-partite*) if its vertices can be t-colored so that every hyperedge has no two vertices of the same color; in other words, it is "t-partite" with vertices partitioned into t parts so that every hyperedge has at most one vertex from each part. Note that t-strong coloring is equivalent to t-coloring the graph obtained by converting each hyperedge into a clique.

Compared to graph coloring, the situation for hardness results for hypergraph coloring is much better. We know that it is NP-hard to color a 2-colorable 3-uniform hypergraph with any constant number of colors [7], and a recent line of work has led to quasi-NP-hardness

[1] The applicability of this result to graphs with small chromatic number seems to have been somewhat overlooked in the literature.

[2] Note that the NP-hardness of 6-coloring 4-colorable graphs would immediately follow from the (as yet unknown) NP-hardness of 5-coloring a 3-colorable graph by adding a new vertex adjacent to all nodes in the graph.

of coloring 2-colorable n-vertex hypergraphs of $O(1)$-uniformity with $\exp(\Omega(\log^{0.1-o(1)} n))$ colors [5, 11, 24, 27, 20], which is approaching the ballpark of polynomially many colors needed by current algorithms.

The 2-coloring problem is easy on hypergraphs $H = (V, E)$ which admit balanced partial colorings. Namely, if there are subsets $A, B \subset V$ such that for each $e \in E$, $|e \cap A| = |e \cap B|$, then one can efficiently find a 2-coloring of H that leaves no hyperedge monochromatic [25]. In particular, a t-uniform t-partite hypergraph, is easy to 2-color. However, even a slight relaxation of the perfect balance condition seems to render 2-coloring intractable. For example, with the promise that there is a near-balanced 2-coloring, finding a 2-coloring without monochromatic edges is still NP-hard [3], and further even c-coloring is NP-hard for any constant c [14].

It will be really interesting to establish further powerful hardness results that show in some formal sense that 2-coloring is hard unless the perfect balance promise is met. Towards this end, the following ultra-strong conjecture postulates that the generally believed hardness of $O(1)$-coloring 3-colorable graphs extends to all strongly colorable hypergraphs with one more color than uniformity (i.e., just beyond the case of a perfectly balanced strong coloring).

▶ **Conjecture 1.2.** *For all $k, c \geq 2$, $(k, c) \neq (2, 2)$, given a $k+1$-strongly colorable k-uniform hypergraph, it is NP-hard to find a c-coloring of its vertices that leaves no hyperedge monochromatic.*

Note that a k-uniform hypergraph that is strongly colorable with $k+1$ colors is also 2-colorable, so the problem in the above conjecture makes sense for any $c \geq 2$. Note the conjecture would immediately yield as a corollary the NP-hardness of telling if a graph G has $\chi(G) \leq t$ or $\chi(G) > c$ for all $c, t \geq 3$, so in this form the conjecture might be well beyond current techniques. However, proving it for $c = 2$ would already be very interesting and this challenge might be within reach by developing more sophisticated analysis tools in the broader framework employed in this paper.

In this work, we prove the following hardness result for 2-coloring strongly colorable hypergraphs, which is the first such result for any promise of strong coloring that implies 2-colorability. Note that a k-uniform hypergraph that is t-strongly colorable for $t \leq 2k-2$ is also 2-colorable (as one can partition the $2k-2$ colors into two groups of $k-1$ and each hyperedge must have colors from both groups).

▶ **Theorem 1.3.** *For $k \geq 3$, given a k-uniform hypergraph, it is NP-hard to tell if it is $\lceil \frac{3k}{2} \rceil$-strongly colorable or if it is not 2-colorable. Further, for $k = 3, 4$, it is NP-hard to 2-color a $k+1$-strongly colorable k-uniform hypergraph.*

The proofs of this theorem can be found in Sections 3.1.1, 3.3, 4, and A.2. In addition, in Appendix A.3, using a simple Fourier-analytic argument, we note the hardness of a variant of $[k, k+1, 2]$, k-odd, in which the sought after two-coloring must be *balanced* (have discrepancy 1) — note that such a balanced 2-coloring *exists* if the hypergraph is $k+1$-partite.

1.1 Prior related work

Towards describing the previous related results in a compact and easy to reference manner, we introduce the following expressive notation, which we will also use in the body of the paper. A k-uniform hypergraph is said to be *t-rainbow colorable* (for some $t \leq k$) if its vertices can be t-colored so that every hyperedge has vertices of every color (note that 2-rainbow colorability is the same as 2-colorability, and for larger t the notion gives a more structured coloring).

▶ **Definition 1.4.** Let $t, k, c \geq 2$ be positive integers. Define $[k, t, c]$-*coloring* to be the following decision problem: Given a be a k-uniform hypergraph H, distinguish between the following two cases.

- YES: If $t < k$, G is t-rainbow colorable; or if $t \geq k$, G is t-strongly colorable. (Note that when $t = k$, t-rainbow and t-strong colorability are the same notion.)
- NO: H is not c-colorable.

Note that when $k = 2$, this is the well-known problem of deciding whether a graph can be colored with at most t colors or requires more than c colors. The algorithm to 2-color a hypergraph in the presence of a balanced partial coloring [25] shows that $[t, t, 2]$-coloring is polynomial time solvable for all $t \geq 2$. The known results on the complexity of $[k, t, c]$-coloring are tabulated below. We will not discuss rainbow coloring further in the paper, but include it in the table below, which also includes two conjectures that 2-coloring is hard if the t-uniform t-strong/rainbow colorability is relaxed for either strong/rainbow coloring. The table does not include algorithmic results for graph/hypergraph coloring where the number of colors used is a function of the number of vertices, or recent hardness results which show hardness of hypergraph coloring with super-polylogarithmically many colors.

Problem	Parameters	Known Hardness	References
Graph coloring	$[2, t, 2t-5]$	NP-hard	[9]
	$[2, t, t+2\lfloor \frac{t}{3} \rfloor - 1]$	NP-hard	[21, 13]
	$[2, t, 2^{2^{\Omega(t^{1/3})}}]$, large t	NP-hard	[19]
	$[2, t, c]$, $c \geq t \geq 3$	UG-variant-hard	[6, 8]
	$[2, t, 2t-2]$	NP-hard	this paper
k-uniform hypergraph coloring	$[k, k, 2]$	in P	[25, 2]
	$[k, 2, c]$, $k \geq 4$, $c \geq 2$	NP-hard	[12]
	$[3, 2, c]$, $c \geq 2$	NP-hard	[7]
t-strong hypergraph coloring	$[3, 4, 2], [4, 5, 2], [\lfloor \frac{2t}{3} \rfloor, t, 2]$, $t \geq 6$	NP-hard	this paper
	$[t-1, t, 2]$, $t \geq 6$	NP-hard (conjectured)	this paper
t-rainbow coloring	$[2t, t, c]$, $t \geq 2$, $c \geq 2$	NP-hard	[14]
	$[t+1, t, 2]$, $t \geq 3$	NP-hard (conjectured)	this paper

Note that we prove the hardness of $[3, 4, 2]$ and $[4, 5, 2]$ coloring separately, for $t \geq 6$, the challenge of proving hardness of $[t-1, t, 2]$ coloring remains open.

We believe that our techniques can also be used to show that $[4, 3, 2]$-coloring and some other problems in the setting of rainbow coloring are NP-hard, but for simplicity and a focused presentation we decided to restrict our study to strong coloring in this version of the paper.

1.2 Techniques

Previous hardness results for approximate coloring of graphs with chromatic number bounded by a constant t fall into three categories:

- NP-hardness for small t, e.g. the $2t-5$-coloring hardness in [9], or the hardness of 4-coloring for $t = 3$: these are based on clever ad hoc reductions from some NP-hard coloring/independent set exact optimization problem (in [21] an approximation version was needed, but the later proof in [13] required only hardness of exact independent set).
- NP-hardness of $f(t)$-coloring for large constant t, such as $f(t) = t^{\Omega(\log t)}$ [22] or the current record $f(t) = \exp(\Omega(t^{1/3}))$ [19]: these are based on designing a PCP with very good query vs. soundness error trade-off and reducing to graph coloring via the FGLSS graph.

These results also show that finding an independent set of density $1/f(t)$ is NP-hard, but they don't kick in until t is reasonably large.

- Hardness of $O(1)$-coloring for $t = 3, 4$ based on variants of the Unique Games Conjecture [6]: these design a 2-query verifier checking the Not-Equal predicate that directly corresponds to graph coloring, and the soundness analysis, which shows that there is no large independent set, relies on appropriate invariance principles. The results showing hardness of $O(1)$-coloring hypergraphs [12, 17, 23, 7] also proceed along this route, but since the Not-All-Equal predicate makes more than two queries, the PCP can be analyzed unconditionally using Fourier analytic tools of the sort pioneered by Håstad [16].

The primary method used to obtain the hardness results in this work departs from the above approaches. We treat coloring as a constraint satisfaction problem (CSP), and our approach is inspired by techniques in the CSP dichotomy literature, where NP-hardness emerges due to the lack of non-dictator "polymorphisms" for the predicate. A polymorphism gives a way to combine several assignments satisfying the predicate into another satisfying assignment. Formally, for an arity k predicate $P \subseteq D^k$ over domain D, a polymorphism for P is a function $f : D^L \to D$ (for some arity L) such that for all $a_1, a_2, \ldots, a_L \in P$, applying f coordinate-wise to the i'th coordinates of $a_1, \ldots, a_L$ for $1 \leq i \leq k$, yields $b \in D^k$ that also belongs P. The dictator functions $f(z) = z_j$ for $j = 1, \ldots, L$ are trivially polymorphisms. If there are no other polymorphisms, then the associated CSP is NP-hard (this connection is folklore, and is mentioned in [4] in a more algebraic language). For instance, if $P \subseteq \mathbb{Z}_3^2$ (for domain $\mathbb{Z}_3 = \{0, 1, 2\}$) is the predicate $\{(x, y) \mid x \neq y\}$, then the only polymorphisms are dictators (this is a nice exercise, and we will prove stronger forms of this for our results). As a result, the associated CSP, which is simply graph 3-colorability, is NP-hard.

Since we seek hardness even when one is allowed more colors, we work in the framework of "weak polymorphisms" from the recent work [3] on hardness of satisfiability even when a near-balanced satisfying assignment exists. Here, the objects of study are relaxations of polymorphisms that map assignments satisfying a predicate P into those that satisfy a *weaker* predicate Q. For instance, to show hardness of 4-coloring 3-colorable graphs, we study functions $f : \mathbb{Z}_3^L \to \mathbb{Z}_4$ satisfying $f(x) \neq f(y)$ whenever $x_i \neq y_i \ \forall i \in \{1, 2, \ldots, L\}$ (in other words, we study 4-colorings of a *dictatorship gadget graph* with vertex set $\mathbb{Z}_3^L$ where two nodes are adjacent precisely when they differ in every coordinate). With 4 colors we can no longer say that f must depend on only coordinate — indeed, we can start with a dictator 3-coloring and corrupt it by recoloring any independent set with the 4'th color. We prove that in fact this is the only thing that can happen — for some $c \in \mathbb{Z}_4$, f restricted to $\mathbb{Z}_4^L \setminus f^{-1}(c)$ is a dictator. For t-colorable graphs, we prove a similar statement classifying functions $f : \mathbb{Z}_t^L \to \mathbb{Z}_{2t-2}$ as comprising of a dictator function for t colors corrupted with $t - 2$ independent sets.

Our proof of Theorem 1.1 follows the common paradigm of reducing from Label Cover, with dictatorship gadgets at each node and cross edges testing the projection constraints. However, our analysis ensures that one can decode, based on a $(2t - 2)$-coloring of the resulting graph, a *unique* label to each vertex that satisfies *all* the label cover constraints. Therefore, as a starting point, we only need the NP-hardness of deciding if a Label Cover instance is satisfiable, and do *not* need a gap version based on PCPs. For the results in [3], the functions which satisfy the dictatorship test are juntas which depend on few variables. This requires starting the reduction from a gap version of Label Cover, as the decoding of labels is not unique. On the other hand, the functions which pass the dictatorship test in [3] are either exact juntas or very close to one, which interfaces nicely with the Label Cover reduction. The challenge in our setting is that the characterization reveals a dictator

function corrupted with a *large* amount of "noise." This is because we have to test functions $f : \mathbb{Z}_t^L \to \mathbb{Z}_c$ with a larger range (with $c > t$, for graph coloring), and for hypergraph coloring, the weak 2-coloring predicate is much weaker than the strong t-coloring promise. For example, for our hardness result for $(2t-2)$-coloring t-colorable graphs, the dictator could be corrupted on almost a $1 - 2/t$ fraction of the hypercube by $t-2$ independent sets. However, the non-noisy portion has a nice structure which helps ensure that the decoded dictatorial coordinate is unique, and further satisfies the projection constraints in the Label Cover instance.

We also abstract a notion of robust decoding of a dictatorship test, which makes the interface with Label Cover more modular, and might help with future reductions based on dictatorship tests.

1.3 Discussion and Limitations

Although the techniques developed produce some new hardness results, there are technical limitations which prevent us from proving better hardness results. For graph coloring, there seem to be fundamental barriers preventing our robust decoding framework from being extended to $[2, t, 2t]$. The primary challenge is that many colorings of the $[2, t, 2t]$ dictatorship test involve nontrivial dependence in multiple coordinates. For example, consider $f_1 : \mathbb{Z}_t^L \to \mathbb{Z}_{2t}$ and $f_2 : \mathbb{Z}_t^L \to \mathbb{Z}_{2t}$ defined by $f_1(x) = x_1$ and $f_2(x) = x_2 + t$. Since the colorings of f_1 and f_2 use separate color sets, any 'interleaving' $f : \mathbb{Z}_t^L \to \mathbb{Z}_{2t}$, for which we choose $f(x) = f_1(x)$ or $f(x) = f_2(x)$ arbitrarily for each x, is a valid coloring of this dictatorship test, too. Furthermore, in Appendix B, we formalize this intuition, we show that there exists no robust decoder for the $[2, 3, 6]$-coloring gadget, which implies that our current methods cannot be used directly to show the NP-hardness of $[2, 3, 6]$-coloring. For similar but more subtle reasons, it also seems likely that no robust decoder exists for the $[2, 3, 5]$-coloring gadget either.

On possible remedy to this technical challenge would be to use a stronger variant of Label Cover known as *smooth* Label Cover. In smooth Label Cover, the edges and projection maps are guaranteed to have pseudorandom properties, allowing for weaker inner verifiers to obtain NP-hardness results. This variant of label cover has been able to prove the NP-hardness of approximation of problems for which the basic variant does not appear to suffice, (e.g., [23, 15, 18, 14]). Currently though, smooth Label Cover does not seem to be sufficient in itself to overcome these technical challenges.

On the other hand, to generalize hypergraph coloring, the primary challenge appears to be the opposite problem. For certain instances, such as $[5, 6, 2]$-coloring, we have a conjectured robust decoder which interfaces well with multipartite Label Cover, but at this time we are unable to determine a combinatorial proof that the robust decoder captures all colorings of the $[5, 6, 2]$ dictatorship test. We conjecture, albeit less confidently, that the situation is similar for $[t-1, t, 2]$-coloring for $t \geq 6$

1.4 Paper Organization

Section 2 constructs the dictatorship gadgets and formally defines the notion of a *robust decoder* of a gadget. Section 3 combinatorially proves the existence of robust decoders for a variety of gadgets. Section 4 uses label cover reductions similar to that of [3] to prove the main theorems. Appendix A contains proofs omitted from Section 3, including a combinatorial classification of the $[4, 5, 2]$-dictatorship test. Appendix B shows that our techniques cannot directly obtain the NP-hardness of 6-coloring a 3-colorable graph.

2 Preliminaries

2.1 The [k,t,c]-coloring Gadget

Adapting the techniques of [3], to prove hardness results we use a label cover reduction with a combinatorial gadget as an inner-verifier long code test. Generalizing the Boolean hypercube, we construct our long code with the *tensor product* of hypergraphs (e.g., [1]).

▶ **Definition 2.1.** Let $G = (V_G, E_G)$ and $H = (V_H, E_H)$ be a k-uniform hypergraphs. The *tensor product* $G \otimes H$ of G and H is the k-uniform hypergraph on vertex set $V_G \times V_H$ such that for all $(g_1, \ldots, g_k) \in E_G$ and $(h_1, \ldots, h_k) \in E_H$ and for all permutations $\sigma : [k] \to [k]$, $((g_1, h_{\sigma(1)}), \ldots, (g_k, h_{\sigma(k)}))$ is an edge of $G \otimes H$.

We let $\otimes^n G$ to denote the tensor product of n copies of G. The most common graph we will be taking the tensor product of is the complete k-uniform hypergraph on t vertices (when $k \leq t$), which we denote K_t^k. We identify the vertices of K_t^k with $\mathbb{Z}_t$ and the edges with k-element subsets of $\mathbb{Z}_t$.

▶ **Definition 2.2** (Dictatorship test gadget)**.** Let $k, t, c \geq 2$ be positive integers such that $k \leq t$ $c \geq t/(k-1)$ and let $L \geq 1$ be an integer. The *dictatorship gadget for* $[k, t, c]$ *on* L *labels* is the k-uniform hypergraph $\otimes^L K_t^k$, the vertices of which we identify with $\mathbb{Z}_t^L$. A *valid coloring* of the dictatorship gadget is a function $f : \mathbb{Z}_t^L \to \mathbb{Z}_c$ such that for all k element subset $S \subseteq \mathbb{Z}_t^L$ which corresponds to an edge of $\otimes^L K_t^k$, $|\{f(x) : x \in S\}| \geq 2$. If f is a valid coloring, then we say that f *satisfies the* $[k, t, c]$*-coloring gadget.*

The constraint $c \geq t/(k-1)$ guarantees that a k-uniform, t-strongly colorable hypergraph has a c-coloring. Note that if $c < t/(k-1)$, then K_t^k is not c-colorable.

We can identify the hyperedges of $\otimes^L K_t^k$ with the k-tuples $(x^{(1)}, \ldots, x^{(k)})$ of $\mathbb{Z}_t^L$ such that for all $i \in [L] = \{1, 2, \ldots, L\}$, $x_i^{(1)}, \ldots, x_i^{(k)}$ are all distinct. Since it is tedious to refer to the underlying graph of a gadget coloring, we formulate a simple syntactic way to check if f satisfies the $[k, t, c]$-coloring gadget. First, we define the notion of being *disjoint* which captures the idea of strong coloring.

▶ **Definition 2.3.** Let $L \geq 1$. A subset $S \subseteq \mathbb{Z}_t^L$ is *disjoint* if $|S| \leq t$ and for all $i \in [L], |\{x_i : x \in S\}| = |S|$. Similarly, we say that $x, y \in \mathbb{Z}_t^L$ are *disjoint* if $\{x, y\}$ is a disjoint subset.

It is easy to verify that the following definition of a coloring of the dictatorship gadget is equivalent.

▶ **Proposition 2.4.** *Let* $k, t, c \geq 2$ *and* $L \geq 1$ *be positive integers such that* $k \leq t$*,* $c \geq t/(k-1)$*. Let* $f : \mathbb{Z}_t^L \to \mathbb{Z}_c$ *be a function. We have that* f *satisfies the* $[k, t, c]$*-coloring gadget if and only if for all* $S \subset \mathbb{Z}_t^L$ *such that* $|S| = k$ *and* S *is disjoint, we have that* $|\{f(x) : x \in S\}| \geq 2$*.*

In our label cover reduction (see Lemma 4.5), a $[k, t, c]$-coloring gadget will be a long code test for a specific vertex of the label cover instance. To represent the edges, we need to construct a $[k, t, c]$-coloring *co-gadget*. This co-gadget is analogous to the edge constraints of [3].

▶ **Definition 2.5.** Let k, t, c, L be positive integers such that $k \leq t$, $c \geq t/(k-1)$. Let $f_1, f_2 : \mathbb{Z}_t^L \to \mathbb{Z}_c$ be functions. We say that $\{f_1, f_2\}$ satisfy the $[k, t, c]$ *co-gadget* if for all disjoint $A \subseteq \mathbb{Z}_t^L$ such that $|A| = k$, and for all partitions[3] $A = A_1 \cup A_2$,

$$|\{f_1(x) : x \in A_1\} \cup \{f_2(x) : x \in A_2\}| \geq 2.$$

[3] Some sets in the partition may be empty.

Notice that if f and g satisfy the $[k,t,c]$ co-gadget, then f and g must both satisfy the $[k,t,c]$ gadget.

2.2 Decoding of Gadgets

With the dictatorship gadgets formulated, we move on to define what it means to *decode* the coloring of a gadget. Toward this, we need to formally define what *dictators* and *juntas* are.

▶ **Definition 2.6** (Dictators)**.** A function $f : \mathbb{Z}_t^L \to \mathbb{Z}_c$ is a *dictator* if there exists $i \in [L]$ and $g : \mathbb{Z}_t \to \mathbb{Z}_c$ such that $f(x) = g(x_i)$ for all $x \in \mathbb{Z}_t^L$.

▶ **Definition 2.7** (Juntas)**.** A function $f : \mathbb{Z}_t^L \to \mathbb{Z}_c$ is a *ℓ-junta* if there exists a $S \subseteq [L]$ with $|S| = \ell$ and $g : \mathbb{Z}_t^S \to \mathbb{Z}_c$ such that $f(x) = g(x|^S)$ for all $x \in \mathbb{Z}_t^L$, where $x|^S$ is the restriction of x to entries indexed by S.

If f satisfies the $[k,t,c]$ gadget and is a dictator we would like to *decode* f into some small subset $S \subseteq [L]$ of coordinates which dictate f's behavior the most. In general though, f is not necessarily a dictator or an ℓ-junta for small ℓ, but it will often be quite close to one. This motivates the following definition.

▶ **Definition 2.8.** For a fixed choice of $k,t,c \geq 2$, A *decoder* is a family[4] of functions $\mathsf{Dec} = \{\mathsf{Dec}^L_{[k,t,c]} : (\mathbb{Z}_t^L \to \mathbb{Z}_c) \to \mathcal{P}([L]) : L \in \mathbb{N}\}$ satisfying the following properties.

- (nontrivial) For all $f : \mathbb{Z}_t^L \to \mathbb{Z}_c$ satisfying the $[k,t,c]$ gadget, $\mathsf{Dec}(f) \neq \emptyset$.
- (sensible) If f depends on the coordinates $S \subseteq [L]$, then $\mathsf{Dec}(f) \subseteq S$. In particular, if f is dictated by the ith coordinate, then $\mathsf{Dec}(f) = \{i\}$.
- (compatible) For all pairs $f, g : \mathbb{Z}_t^L \to \mathbb{Z}_c$, which satisfy the $[k,t,c]$ co-gadget, $\mathsf{Dec}(f) \cap \mathsf{Dec}(g) \neq \emptyset$.
- (bounded) There exists a constant $C = C(k,t,c)$ independent of L, such that $|\mathsf{Dec}(f)| \leq C$ for all choices of f.

We say that $\mathsf{Dec}(f)$ is the *decoding* of f.

Due to the technical details of our label cover reduction, to obtain NP-hardness results we need our decoder to have one additionally property, that the decoding of f needs to compose nicely with *projections*.

▶ **Definition 2.9.** Let $f : \mathbb{Z}_t^L \to \mathbb{Z}_c$ be a function. Let $\pi : [L] \to [L]$ be a projection. Define the *restriction of f with respect to π*, denoted $f\square\pi : \mathbb{Z}_t^L \to \mathbb{Z}_c$ to be the unique map satisfying

$$(f\square\pi)(x) = f(y) \text{ where } y_i = x_{\pi(i)} \text{ for all } i \in [L].$$

In other words, $f\square\pi$ applies f after 'copying' coordinates in the image of π to coordinates with that projection.

Note that if f satisfies the $[k,t,c]$ gadget, then $f\square\pi$ also satisfies the $[k,t,c]$ gadget for all π.

▶ **Definition 2.10.** We say that a decoder $\mathsf{Dec} = \mathsf{Dec}_{[k,t,c]}$ is *robust* if for all $f : \mathbb{Z}_t^L \to \mathbb{Z}_c$ satisfying the $[k,t,c]$ gadget and all projections $\pi : [L] \to [L]$, $\mathsf{Dec}(f\square\pi) \subseteq \pi[\mathsf{Dec}(f)]$.

[4] For $f : \mathbb{Z}_t^L \to \mathbb{Z}_c$, we use $\mathsf{Dec}(f)$ or $\mathsf{Dec}_{[k,t,c]}(f)$ as a shorthand for $\mathsf{Dec}^L_{[k,t,c]}(f)$.

3 Colorings of Dictatorship Gadgets

Now that have constructed our dictatorship gadgets/co-gadgets and defined the notion of a robust decoder, we proceed to demonstrate to prove that a number of $[k, t, c]$-coloring gadgets indeed have robust decoders.

3.1 Small Examples

To better understand gadget colorings, we often examine subsets $S \subseteq \mathbb{Z}_t^L$ such that no pair of elements of S are disjoint. We call such an S an *independent set*. For any $S \subseteq \mathbb{Z}_t^L$ we let the *clique number* $\omega(S)$ denote the size of the largest disjoint subset of S. The following fact relates the clique number of S to the density of S in $\mathbb{Z}_t^L$.

▶ **Claim 3.1.** *For any $S \subseteq \mathbb{Z}_t^L$,*

$$\frac{|S|}{|\mathbb{Z}_t^L|} \leq \frac{\omega(S)}{t} \tag{1}$$

with equality if and only if the indicator function for S is a dictator.

▶ Remark. For independent sets ($\omega(S) = 1$), this claim is well-known (e.g., [10]). See [1] for a particularly elegant proof involving Fourier analysis.

Proof. See Appendix A. ◀

3.1.1 Strong coloring: [3,4,2] gadget

Using Claim 3.1, we may easily classify the colorings of $[3, 4, 2]$ gadget. Recall that this gadget corresponds to the 2-coloring of 3-uniform 4-strong colorable graphs.

▶ **Claim 3.2.** *Let $f : \mathbb{Z}_4^L \to \mathbb{Z}_2$ satisfy the $[3, 4, 2]$-coloring gadget. Then, there exists $i \in [n]$ and $S \subset \mathbb{Z}_4$ such that $|S| = 2$ and for all $x \in \mathbb{Z}_4^L$, $f(x) = 0$ iff $x_i \in S$.*

Proof. From the definition of the gadget $\omega(f^{-1}(0)), \omega(f^{-1}(1)) \leq 2$. Thus, $|f^{-1}(0)|, |f^{-1}(1)| \leq \frac{1}{2}|\mathbb{Z}_4^L|$. Since $\mathbb{Z}_4^L = f^{-1}(0) \cup f^{-1}(1)$, we must have that $|f^{-1}(0)| = |f^{-1}(1)| = \frac{1}{2}|\mathbb{Z}_4^L|$. Thus, the indicator function for $|f^{-1}(1)|$ must be a dictator in the ith coordinate, implying that f is a dictator in the ith coordinate. This implies the conclusion of the claim. ◀

Since any f which satisfies this gadget must be a dictator, the natural choice for a decoder $\mathsf{Dec}_{[3,4,2]}$ is to decode the index of the dictatorial coordinate.

▶ **Claim 3.3.** *There exists a robust decoder $\mathsf{Dec} = \mathsf{Dec}_{[3,4,2]}$ for $[3, 4, 2]$-coloring.*

Proof. Let $\mathsf{Dec}(f) = \{i\}$, where i is the coordinate that f is a dictator. Clearly Dec is nontrivial, sensible, and bounded. To establish that Dec is compatible, assume for sake of contradiction f_1 and f_2 satisfy the $[3, 4, 2]$-coloring co-gadget, but f_1 and f_2 are dictated by different coordinates. Without loss of generality, we may assume that f_1 is dictated by the first coordinate, and f_2 is dictated by the second coordinate. Furthermore, we may assume without loss of generality that $f_1(x) = 0$ if and only if $x_1 \in \{0, 1\}$ and that $f_2(x) = 0$ if and only if $x_2 \in \{0, 1\}$. But then, we may select $A_1 = \{(0, 2, 0, 0, \ldots, 0), (1, 3, 1, 1, \ldots, 1)\}$ and $A_2 = \{(2, 0, 2, 2, \ldots, 2)\}$ such that $A_1 \cup A_2$ is disjoint, but $f_1(A_1) \cup f_2(A_2) = \{0\}$, violating the $[3, 4, 2]$-coloring co-gadget, a contradiction. Thus, Dec is indeed compatible. Therefore, Dec is a decoder.

Note that if f is a dictator in coordinate i, then for any projection $\pi : [L] \to [L]$, $f \square \pi$ is a dictator in coordinate $\pi(i)$. Thus, $\mathsf{Dec}(f \square \pi) \subseteq \pi(\mathsf{Dec}(f))$, establishing that Dec is robust, as desired. ◀

If we combine Claim 3.3 with Lemma 4.5, we have that $[3, 4, 2]$-coloring is NP-hard.

In Appendix A.2, we give a combinatorial classification of the $[4, 5, 2]$ dictatorship gadget.

3.1.2 Graph coloring: [2,3,4] gadget

Next, we classify the $[2, 3, 4]$ gadget which corresponds to problem of coloring a 3-colorable graph with 4 colors. This will form the base case for our more general $[2, t, 2t - 2]$-hardness result in Section 3.2. The following lemma is a key ingredient in the proof of our main result.

▶ **Lemma 3.4.** *Let $f : \mathbb{Z}_3^L \to \mathbb{Z}_4$ satisfy the $[2, 3, 4]$ gadget. Then, there exists $a \in \mathbb{Z}_4$ such that f restricted to $\mathbb{Z}_3^L \setminus f^{-1}(a)$ is a dictator.*

Proof. We say that three points $x, y, z \in \mathbb{Z}_3^L$ form an *axis-parallel line* if x, y, z differ in exactly one coordinate.

▶ **Claim 3.5.** *There does not exist $x \in \mathbb{Z}_3^L$ and lines $\{x, y, z\}$ and $\{x, y', z'\}$ such that both lines are axis-parallel to the coordinate axes and each line takes on 3 distinct values with respect to f.*

For ease of notation, when referring to a subset of $\mathbb{Z}_3^L$ we may concatenate digits to indicate an ordered tuple. For example,

- 012 is the ordered tuple $(0, 1, 2)$.
- $12\{0, 1\}^2$ is the set $\{(1, 2, 0, 0), (1, 2, 0, 1), (1, 2, 1, 0), (1, 2, 1, 1)\}$
- $\{12, 21\} \times \{0\}$ is $\{(1, 2, 0), (2, 1, 0)\}$.

Proof. If $L = 1$, the claim is trivial. Assume for sake of contradiction that such an x exists. Without loss of generality, such x is $0 \ldots 0$ the two axis-parallel lines differ in the first and second coordinates. Thus, we may assume without loss of generality that

$$f(0 \ldots 0) = 0 \qquad f(10 \ldots 0) = 1 \qquad f(20 \ldots 0) = 2 \ .$$

We may also assume without loss of generality that

$$f(010 \ldots 0) = 1.$$

Since 01 and 10 are disjoint, $L = 2$ is impossible. Now we have two cases.

Case 1, $f(020 \ldots 0) = 2$. Notice that we may then deduce that

$$f(12\{1, 2\}^{L-2}) = 3 \qquad f(21\{1, 2\}^{L-2}) = 3 \ .$$

Since there are disjoint elements in $\{12, 21\} \times \{1, 2\}^{L-2}$, we have a contradiction.

Case 2, $f(020\ldots0) = 3$. Now, we may deduce that

$$f(11\{1,2\}^{L-2}) = 1 \tag{2}$$
$$f(12\{1,2\}^{L-2}) = 3 \tag{3}$$
$$f(21\{1,2\}^{L-2}) = 2. \tag{4}$$

Notice that this implies that

$$f(00\mathbb{Z}_3^{L-2}) = 0. \tag{5}$$

From this, we may deduce that

$$f(\{22\} \times \mathbb{Z}_3^{L-2}) \subseteq \{2,3\}. \qquad \text{[using (2) and (5)]}$$

If there exists, $x, y \in \{0,1\}^2 \times \mathbb{Z}_3^{L-2}$ such that $f(x) = 2$ and $f(y) = 3$, then there is some $z \in \{22\} \times \mathbb{Z}_3^{L-2}$ that is disjoint from both x and y, a contradiction. Notice then, due to symmetry, we may assume without loss of generality that

$$f(x) \neq 2, x \in \{0,1\}^2 \times \mathbb{Z}_3^{L-2}. \tag{6}$$

Therefore, we have that

$$f(01\mathbb{Z}_3^{L-2}) \subseteq \{0,1\}. \qquad \text{[using (3) and (6)]}$$

Thus, similar logic, there cannot be $x, y \in \{10\} \times \mathbb{Z}_3^{L-2}$ such that $f(x) = 0$ and $f(y) = 1$. Since $f(10\ldots0) = 1$, we have that

$$f(x) \neq 0, x \in \{10\} \times \mathbb{Z}_3^{L-2}. \tag{7}$$

Therefore,

$$f(\{10\} \times \mathbb{Z}_3^{L-2}) \subseteq \{1,3\}. \qquad \text{[using (4) and (7)]}$$

But, this is at odds with $f(020\ldots0) = 3$ and $f(010\ldots0) = 1$, leading to a contradiction. ◀

▶ **Claim 3.6.** *If there exist $x_0, x_1, x_2 \in \mathbb{Z}_3^L$ which each differ in exactly one coordinate (that is, form a line parallel to an axis) and $f(x_0) \neq f(x_1) = f(x_2)$, then f satisfies the conclusion of Lemma 3.4.*

Proof. Assume without loss of generality that $x_0 = 0\ldots0$, $x_1 = 10\ldots0$, and $x_2 = 20\ldots0$, and

$$f(x_0) = f(0\ldots0) = 0 \qquad f(x_1) = f(10\ldots0) = 1 \qquad f(x_2) = f(20\ldots0) = 1\ .$$

From this, we may deduce that

$$f(\{1,2\}^L) \subseteq \{2,3\}. \tag{8}$$

Additionally, any two disjoint points of $\{1,2\}^L$ must take on different values with f. Consider any $y_0, y_1 \in \{1,2\}^L$ which differ in exactly one coordinate but also differ with respect to f:

$$f(y_0) = 2, f(y_1) = 3.$$

Let y_2 be the third point on the axis-parallel line between y_0 and y_1, and let y_0' and y_1' be the points of $\{1,2\}^L$ disjoint from y_0 and y_1, respectively. Thus,

$$f(y_0') = 3, f(y_1') = 2.$$

Since y_2 is disjoint from both y_0' and y_1', we have that

$$f(y_2) \in \{0,1\}.$$

Hence, we have deduced that for any $y_0, y_1 \in \{1,2\}^L$ which differ in exactly one coordinate and also differ with respect to f, the third point on the line between them must also differ with respect to f. Therefore, by Claim 3.5, we necessarily have that for all $y \in \{1,2\}^L$, there is at most one $z \in \{1,2\}^L$ differing with f in exactly one coordinate with $f(y) \neq f(z)$.
From this fact, we can assign a function $g : \{1,2\}^L \to [L] \cup \{\text{None}\}$ such that if $y, z \in \{1,2\}^L$ are neighbors but $f(y) \neq f(z)$ then $g(y) = g(z)$ is the coordinate they differ in. Furthermore, if $g(y) \neq$ None, we claim that for all $w \in \{1,2\}^L$ which differ by y in exactly one coordinate, $g(y) = g(w)$. If it were the case $g(y) \neq g(w)$, let $y_0, w_0 \in \{1,2\}^L$ which differ from y and w in the $g(y)$th coordinate. Then $f(y) = f(w) = f(w_0) \neq f(y_0)$, implying that $g(y_0)$ should be both $g(y)$ (due to y_0)and $g(w)$ (due to w_0), a contradiction.
Since the Hamming graph on $\{1,2\}^L$ connected and f takes on multiple values in $\{1,2\}^L$, we have that g takes on a non-None value at at least one point and since the Hamming graph is connected, we must have that g is a constant function. Thus, f restricted to $\{1,2\}^L$ is a dictator. Let i be the coordinate in which f restricted to $\{1,2\}^L$ is a dictator. And assume without loss of generality due to (8) that

$$x \in \{1,2\}^L, f(x) = 2 \text{ if and only if } x_i = 1.$$

From this and (8), we may deduce that for a general $x \in \mathbb{Z}_3^L$,

$$x_i = 0 \text{ then } f(x) \in \{0,1\}$$
$$x_i = 1 \text{ then } f(x) \in \{0,1,2\}$$
$$x_i = 2 \text{ then } f(x) \in \{0,1,3\}.$$

Notice that if there are $x, y \in \mathbb{Z}_3^L$ (not necessarily disjoint) with $x_i, y_i \neq 0$ such that $f(x) = 0$ and $f(y) = 1$, then we can identify at least one $z \in \mathbb{Z}_3^L$ with $z_i = 0$ such that z is disjoint from x and y, implying that $f(z) \notin \{0,1\}$, a contradiction. Thus, without loss of generality, we can say that $f(x) \neq 1$ if $x_i \neq 0$. Therefore, f restricted to $\mathbb{Z}_3^L \setminus f^{-1}(0)$ is a dictator in the ith coordinate ($f(x_i) = x_i + 1$), as desired. ◀

(Back to proof of Lemma 3.4) Assume for sake of contradiction that there is a counterexample. From Claim 3.6, we know that no three of $y_0, y_1, y_2 \in \mathbb{Z}_3^L$ differing in exactly one coordinate which take on two distinct values with respect to f. But, we also know from Claim 3.5 that no $x \in \mathbb{Z}_3^L$ has two axis-parallel lines through it that take on 3 different values. This implies that for each $x \in \mathbb{Z}_3^L$, there is at most coordinate for which changing x changes the value of f. As in the proof of Claim 3.6, we may construct $g : \mathbb{Z}_3^L \to [L] \cup \{\text{None}\}$ such that if x and x' are neighbors which differ then $g(x) = g(x')$ is the coordinate they differ in. Again, for all $x \in \mathbb{Z}_3^L$ such that $g(x) \neq$ None, we have that all the neighbors (at Hamming distance 1) of x must take on the same value for g. Since f is not constant, g takes on at least one non-None value. Thus by a flood-fill argument, g must be a constant function. Hence, f must be a dictator, implying that there is no counterexample, as desired. ◀

We wait to show that the $[2,3,4]$ gadget has a robust decoder until we establish the generalization for the $[2,t,2t-2]$ gadget.

3.2 Graph coloring: The general case

▶ **Lemma 3.7.** *Let $f : \mathbb{Z}_t^L \to \mathbb{Z}_{2t-2}$ satisfy the $[2, t, 2t-2]$-colorability gadget where $t \geq 3$. Then there exists $S \subset \mathbb{Z}_{2t-2}$ such that $|S| = t-2$ and f restricted to $\mathbb{Z}_t^L \setminus f^{-1}(S)$ is a dictator.*

Since by Claim 3.1, we have that for all $c \in \mathbb{Z}_{2t-2}$, $|f^{-1}(c)| \leq t^{L-1}$. Thus, by 'discarding' only $t-2$ of the colors, we have an understanding of the structure of at least $2/t$ fraction of the coloring. This is enough structure to obtain NP-hardness in our label cover reduction in Section 4.

Proof. We proceed by induction on t. The base case $t = 3$ follows from Lemma 3.4.

▶ **Claim 3.8.** *If $t \geq 4$ and f restricted to $\mathbb{Z}_{t-1}^L$ satisfies the $[2, t-1, 2t-4]$-colorability gadget and we assume the result is true for $t-1$, then f satisfies the conclusion of the lemma.*

Proof. From the inductive hypothesis, we have that there is $S' \subseteq \mathbb{Z}_{2t-4}$ with $|S'| = (t-1) - 2 = t-3$ such that f restricted to $\mathbb{Z}_{t-1}^L \setminus f^{-1}(S')$ is a dictator. Since $(t-1)+(t-3) = 2t-4$, there is a subset $S = \mathbb{Z}_{2t-4} \setminus S' \subseteq \mathbb{Z}_{2t-4}$ of size $t-1$ such that f restricted to $f^{-1}(S) \cap \mathbb{Z}_{t-1}^L$ is a dictator in some coordinate. Assume without loss of generality that $S = \{0, 1, \ldots, t-2\}$. Additionally, we may assume that $f(x) = x_1$ when $x \in f^{-1}(S) \cap \mathbb{Z}_{t-1}^L$. Thus, f in this restricted domain is a dictator in the first coordinate.

Still working in $\mathbb{Z}_{t-1}^L$, we let T_i be the set of colors in the image of f with respect to the set of points where the first coordinate is i. More formally, $T_i = \{f(x) : x \in \mathbb{Z}_{t-1}^L, x_1 = i\}$. Since $T_i \subset \mathbb{Z}_{2t-4}$ and by our assumption $T_i \cap S = \{i\}$, we have that $|T_i| \leq t-2$ for all $i \in \mathbb{Z}_{t-1}$.

As a key part of our inductive step, for each $i \in \mathbb{Z}_{t-1}$, we seek to select a color $c_i \in T_i$ such that for *all* $x \in \mathbb{Z}_t^L$, $f(x) = c_i$ implies that $x_1 = i$. Note that it might be the case that $c_i \neq i$. Assume for sake of contradiction that there exists $i \in \mathbb{Z}_t$ such that for all $c \in T_i$, there is $x^{(c)} \in \mathbb{Z}_t^L$ with $x_1^{(c)} \neq i$ but $f(x^{(c)}) = c$. Since $|T_i| \leq t-2$, there must $z \in \mathbb{Z}_{t-1}^L$ which is disjoint from every element of the set $\{x^{(c)} : c \in T_i\}$. Since we stipulated that $x_1^{(c)} \neq i$ for all $c \in T_i$, we may select that $z_1 = i$. Since $z \in \mathbb{Z}_{t-1}^L$ and $x_1 = i$, by definition of T_i, $f(z) \in T_i$. But, by definition of z, z is disjoint from $x^{(f(z))}$, so $f(z) \neq f(x^{f(z)}) = f(z)$, a contradiction. Thus, for all $i \in \mathbb{Z}_{t-1}$, we can find an *exclusive* color c_i; that is, $f(x) = c_i$ implies $x_1 = i$ for all $x \in \mathbb{Z}_t^L$.

To complete the claim, it suffices to find a color c_{t-1} such that $f(x) = c_{t-1}$ implies $x_1 = t-1$. Let $T_{t-1} = \mathbb{Z}_{2t-2} \setminus \{c_i : i \in \mathbb{Z}_{t-1}\}$. That is, T_{t-1} is set of colors that are not already exclusive. Thus, if $x \in \mathbb{Z}_t^L$ and $x_1 = t-1$, then we must have that $f(x) \in T_{t-1}$. It is easy to see that $|T_{t-1}| = 2t-2-(t-1) = t-1$. Assume for sake of contradiction that an exclusive color c_{t-1} does not exist. Thus, for all $c \in T_{t-1}$, there is $y^{(c)}$ such that $f(y^{(c)}) = c$ but $y_1^{(c)} \neq t-1$. Thus, we may select $z \in \mathbb{Z}_t^L$ disjoint from every element of $\{y^{(c)} : c \in T_{t-1}\}$. Furthermore, since $y_1^{(c)} \neq t-1$ for all $c \in T_{t-1}$, we can let $z_1 = t-1$. By choice of z, we have that $f(z) \notin T_{t-1}$. Thus, $f(z) \in \mathbb{Z}_{2t-2} \setminus T_{t-1} = \{c_i : i \in \mathbb{Z}_{t-1}\}$. Thus, $f(z) = c_i$ for some $i \in \mathbb{Z}_{t-1}$, implying that $z_1 = i$, a contradiction. Therefore, there is $c_{t-1} \in T_{t-1}$ such that $f(x) = c_{t-1}$ implies that $x_1 = t-1$, as desired.

Hence, f restricted to $f^{-1}(\{c_i : i \in \mathbb{Z}_t\})$ is a dictator, as desired. ◀

Consider any axis-parallel line ℓ of $\mathbb{Z}_t^L$. If there exists $x, y, z \in \ell$ such that $f(x) \neq f(y) = f(z)$, then the $(t-1)^L$ subgrid disjoint from x cannot have either $f(x)$ or $f(y)$ in the image of f. Thus this subgrid satisfies the $[2, t-1, 2t-4]$ gadget and then we are done by Claim 3.8.

Thus, every axis-parallel line must be entirely distinct or entirely the same. Next, we seek to show that any counterexample cannot contain two perpendicular axis-parallel lines which take on entirely distinct values. Without loss of generality, we may assume that for all $i \in \mathbb{Z}_t$, $f(i0\dots0) = i$. We may also assume that $f(010\dots0) = c \in \{1,\dots t-1\}$ since there are only $t-2$ values in $\{t,\dots,2t-3\}$ and $f(0\dots0) = 0$. Now notice that $f(c2\{1,\dots,t-1\}^{L-2}) \subseteq \{t,t+1,\dots,2t-3\}$ Thus, this $(t-1)^{L-2}$ grid can only take on $t-2$ values. In order for every axis-parallel line to be completely the same or completely distinct, we must have that $f(c2\mathbb{Z}_t^{L-2}) = c_2$ for some $c_2 \notin \mathbb{Z}_t$. Similarly, $f(c\ell\mathbb{Z}_t^{L-2}) = c_\ell$ for some $c_\ell \notin \mathbb{Z}_t$. Since $f(c0\dots0) = c$, we cannot have any two c_i be equal. Thus, we may assume without loss of generality that $c_k = t+k-2$ for all $k \geq 2$. Since $t \geq 4$, we must have that $f(c1\{1,\dots,t-1\}^{L-2})$ cannot take on any element in $\{t,\dots,2t-3\}$ without forcing a non-distinct, non-homogeneous axis-parallel line. Thus, $f(c1\{1,\dots,t-1\}^{L-2})$ can only take on the value c. Since every axis-parallel line through at least two points in $c1\{1,\dots,t-1\}^{L-2}$ must take on the value c, by an inductive argument we can deduce that $f(c1\mathbb{Z}_t^{L-2}) = c$. Hence, $f(c0\dots0) = f(c10\dots0) = c$ but $f(c20\dots0) = c_2 \neq c$, a contradiction. Thus, any counterexample cannot contain two perpendicular axis-parallel lines taking on distinct values.

Clearly f cannot be constant. Thus, there is an least one distinct line. Using an argument quite similar to the one in the proof of Lemma 3.4, for any x in this line, the neighbors of x (those at Hamming distance at most 1 away) must also be on an axis-parallel-line in the same direction. Thus, f is forced to be a dictator, as desired. ◀

Now that we understand the $[2,t,2t-2]$ gadget well, we can now establish the existence of robust decoders.

▶ **Lemma 3.9.** *For all $t \geq 3$, the $[2,t,2t-2]$ gadget has a robust decoder* Dec.

Proof. For any $f : \mathbb{Z}_t^L \to \mathbb{Z}_{2t-2}$, let $\mathsf{Dec}(f) \subseteq [L]$ be the set of coordinates i such that there is a t-element subset $S \subset \mathbb{Z}_{2t-2}$ such that f restricted to $f^{-1}(S)$ is a dictator. We call S a *witness* for i. We now show that our decoder meets all of the conditions of Definitions 2.8 and 2.10.

- nontrivial: From Lemma 3.7, we know that $\mathsf{Dec}(f) \neq \emptyset$.
- sensible: If $i \in \mathsf{Dec}(f)$, let S be a witness for i. For every $x \in f^{-1}(S)$ and all x' such that x' and x only differ in the ith coordinate, $f(x) \neq f(x')$. Thus, f has dependence in the ith coordinate.
- bounded: We claim that $|\mathsf{Dec}(f)| = 1$ always, implying boundedness. Assume that there exist $i \neq j$ such that $i,j \in \mathsf{Dec}(f)$. Let S_i and S_j be the subsets of t colors such that f restricted to $f^{-1}(S_i)$ and $f^{-1}(S_j)$ are dictators in the ith and jth coordinate respectively. Let $A = S_i \cap S_j$. Clearly $|A| \geq |S_i| + |S_j| - |\mathbb{Z}_{2t-2}| = 2$. Let $B_i = \{x_i : f(x) \in A\}$ and $B_j = \{x_j : f(x) \in A\}$. It is easy to see that $|B_i| = |B_j| = |A|$. Now consider $K = \{x \in \mathbb{Z}_t^L : x_i \in B_i, x_j \in B_j\}$. It is easy to see that $|K|/|Z_t^L| = |B_i||B_j|/t^2 = |A|^2/t^2$. By definition of i and j, we can see that K is disjoint from $f^{-1}((S_i \cup S_j) \setminus A)$.
  We claim that for any color $c \in \mathbb{Z}_{2t-2}$, $|f^{-1}(c) \cap K| \leq |K|/|A|$. The proof of this is similar to the proof of Claim 3.1. Without loss of generality, assume that $B_i = B_j = \mathbb{Z}_{|A|}$. We then can see the following is a partition of K into disjoint cliques of size K

$$\bigcup_{x \in K, x_i = 0} \{(x, x+(1,\dots,1),\dots,x+(|A|-1,\dots,|A|-1))\},$$

  where addition in the ith and jth coordinates is modulo $|A|$.

Also, for any $c \in A$, it is apparent that $|f^{-1}(c) \cap K| \leq |K|/|A|^2$. Combining these two facts, $|f(K)| \geq 2|A| - 1$ (the $|A|$ colors in A and the additional $|A| - 1$ colors needed by the bound above). Thus, $2t - 2 \geq |f(K)| + |(S_i \cup S_j) \setminus A| \geq 2|A| - 1 + 2t - 2|A| = 2t - 1$, a contradiction. Thus, $|\mathsf{Dec}(f)| = 1$, as desired.

- compatible: Consider any $f, g : \mathbb{Z}_t^L \to \mathbb{Z}_{2t-2}$ which satisfy the $[2, t, 2t-2]$ co-gadget. We claim that $\mathsf{Dec}(f) = \mathsf{Dec}(g)$. Consider $h : \mathbb{Z}_t^{L+1} \to \mathbb{Z}_{2t-2}$ such that $h(x, 0) = f(x)$ and $h(x, i) = g(x)$ for all $i \in \mathbb{Z}_t \setminus \{0\}$. Since f and g satisfy the $[2, t, 2t-2]$ co-gadget, h satisfies the $[2, t, 2t-2]$ gadget. Thus, we can decode h in a unique coordinate i. Let S be the witness for i of h. If $i = L + 1$, then $h(\mathbb{Z}_t^L \times \{0\}) = f(\mathbb{Z}_t^L)$ can only have $2t - 2 - (t - 1) = t - 1$ colors, a contradiction. Thus, $i \in [L]$. It is easy then to see that S is also a witness for f and g. Thus, due to unique decoding, $\mathsf{Dec}(f) = \mathsf{Dec}(g) = \{i\}$, as desired.
- robust: Consider any projection $\pi : [L] \to [L]$. For $\{i\} = \mathsf{Dec}(f)$, let S our witness. Since f is a dictator with respect to $f^{-1}(S)$, $f \square \pi$ is a dictator in coordinate $\pi(i)$ respect to $(f \square \pi)^{-1}(S)$. Thus, S is a witness of $\pi(i)$ for $f \square \pi$, $\mathsf{Dec}(f \square \pi) = \{\pi(i)\}$. Thus, $\mathsf{Dec}(f \square \pi) \subseteq \pi(\mathsf{Dec}(f))$.

Hence, the $[2, t, 2t-2]$ gadget has a robust decoder. ◀

3.3 [k,⌈3k/2⌉,2] combinatorial characterization

To obtain hardness results for our hypergraph coloring problem, we first prove a characterization of the two-colorings of the strong hypergraph coloring dictatorship gadget.

▶ **Lemma 3.10.** *Let $f : \mathbb{Z}_t^L \to \mathbb{Z}_2$ satisfy the $[k, t, 2]$ gadget, where $\lceil t/2 \rceil + 1 \leq k \leq \lfloor 2t/3 \rfloor$. Then, there exists $i \in [L]$ and dictator-bounding $D_j \subseteq \mathbb{Z}_t$ for all $j \in [L]$ such that*

- *$|D_j| = 2t - 2k + 2$ for all $j \in [L]$*
- *there is a function $g : D_i \to \mathbb{Z}_2$ such that for all $x \in \mathbb{Z}_t^L$ such that $x_j \in D_j$ for all j, $f(x) = g(x_i)$. Furthermore $|g^{-1}(0)| = |g^{-1}(1)| = t - k + 1$.*

To motivate the structure of the proof, we first handle the special case $t = 2k - 2$.

Proof of $t = 2k - 2$ case. We seek to show that f must be a dictator with an even split of 0s and 1s. Assume without loss of generality that

$$f(00\ldots0) = 0, f(10\ldots0) = 1$$

Now, consider f restricted to $S = \{2, \ldots, t-1\} \times \{1, \ldots, t-1\}^{L-1}$. If this is not a dictator in the first coordinate, then we may select axis parallel $x, y \in S$ which are the same in the first coordinate such that $f(x) = 0, f(y) = 1$. It is easy to see that there is a disjoint set of size $t - 3 = 2k - 5$ which is disjoint from all of $\{0\ldots0, 10\ldots0, x, y\}$. Thus are at least $\lceil (t-3)/2 \rceil = k - 2$ points in this disjoint set which are of the same value. Adding to this set either $\{0\ldots0, x\}$ or $\{10\ldots0, y\}$, we have that there is a disjoint set of size k all of which take on the same value with respect to f, a contradiction. Thus, f restricted to S must be a dictator in the first coordinate. In order to avoid any disjoint sets of size k, we must have that f restricted to S has an equal number of 1s and 0s.

Now, take any axis-parallel pair $x, y \in S$ differing in the first coordinate such that $f(x) = 0$ and $f(y) = 1$. Using the same argument (where we replace $00\ldots0$ with x and $10\ldots0$ with y), we have that the set of points disjoint from x and y must be a dictator in the first coordinate. Applying this fact to every such x and y in S, we can see that f restricted to $\{0, 1\} \times \mathbb{Z}_t^{L-1}$ is a dictator in the first coordinate with $f(0\mathbb{Z}_t^{L-1}) = 0$ and $f(1\mathbb{Z}_t^{L-1}) = 1$. Next, if we consider all axis-parallel pairs $x \in \{0\} \times \mathbb{Z}_t^{L-1}, y \in \{1\} \times \mathbb{Z}_t^{L-1}$, we may deduce

Figure 1 Illustration of the proof of Lemma 3.10 in the case $L = 2, k = 4, t = 6$. The grids represent values deduced of $f : \mathbb{Z}_6^2 \to \{0, 1\}$. Left: if $f(00) = f(21) = 0$ and $f(10) = f(22) = 1$, it is impossible to fill in the values of $\{3, 4, 5\}^2$ without forcing a monochromatic 4-uniform hyperedge. Right: if $f(00) = 0$, $f(10) = 1$, and $f : \{2, 3, 4, 5\} \times \{1, 2, 3, 4, 5\} \to \{0, 1\}$ is a 'balanced' dictator in the first coordinate, then in order to avoid a monochromatic 4-uniform hyperedge, $f(0x) = 0$ and $f(1x) = 1$ for all $x \in \{1, 2, 3, 4, 5\}$.

that f restricted to $\{2, \dots, t-1\} \times \mathbb{Z}_t^{L-1}$ is a dictator in the first coordinate. Thus, f is a dictator in the first coordinate with an equal number of 0 and 1s (in order to avoid a $(t/2+1)$-sized monochromatic hyperedge), as desired. ◀

Full argument. We proceed by strong induction on t. Our base case $k = 4, t = 6$ is handled above.

Let $x^1 = 0 \dots 0, y^1 = 10 \dots 0$. Assume without loss of generality that $f(x^1) = 0, f(y^1) = 1$. Now, consider the subgrid $T = \{2, \dots, t-1\} \times \{1, \dots, t-1\}^{L-1}$ which is disjoint from x^1 and y^1.

▶ **Claim 3.11.** *There exists $D'_j \subseteq \mathbb{Z}_t$ $(j \in [L])$ such that $|D'_j| \geq 2t - 2k$ for all j, f restricted to the Cartesian product $\bigtimes_{j \in L} D'_j \subseteq T$ is a dictator in some coordinate ℓ. Furthermore, exactly $t - k$ values of D'_ℓ set f to 0.*

Proof. First, assume we can find axis-parallel $x^2, y^2 \in T$ such that $f(x^2) = 0, f(y^2) = 1$, but the coordinate x^2 and y^2 differ in is not the first coordinate. Without loss of generality assume that $x^2 = 21 \dots 1$ and $y^2 = 221 \dots 1$. Now consider $T' = \{3, \dots t-1\}^L$ which is disjoint from x^1, x^2, y^1, and y^2. Clearly f restricted to T' satisfies the $[k-2, t-3, 2]$ gadget since if f were to have a disjoint set of size $k-2$ in T', either $\{x^1, x^2\}$ or $\{y^1, y^2\}$ could be augmented to yield a disjoint set of size k which is constant with respect to f. Clearly if $\lfloor 2t/3 \rfloor \geq k$ then $\lfloor 2(t-3)/3 \rfloor \geq k-2$. The last thing we need to check to apply the induction hypothesis is that $t - 3 \geq 6$. We can find a disjoint set of size $\lceil (t-3)/2 \rceil$ which is constant with respect to f, so $k - 2 > \lceil (t-3)/2 \rceil$. It is easy to check this is false if $t < 9$ since $k \leq \lfloor 2t/3 \rfloor$, yielding a contradiction. Thus, $t \geq 9$, so $t - 3 \geq 6$ so we may use the induction hypothesis to find D'_j which satisfy the claim.

Now, assume that no such $x^2, y^2 \in T$ exist. Then, f restricted to T is a dictator in the first coordinate. All we need to check is that there are at least $t - k$ choices for the first coordinate so that f restricted to T is equal to 0 (respectively 1) in the first coordinate. If we cannot find $t - k$ choices for, without loss of generality 0, then there are at least $(t-2) - (t-k-1) = k-1$ choices for 1. Since there are at least $t-2$ (which is at least $k-1$) choices in each of the other coordinates, we can select $k-1$ disjoint points which take on the value 1 within T. If we add in y^1, we have a set of k disjoint points which take on the value 1 within T. ◀

Thus, we have that f restricted to $T' = \bigtimes_{j \in [L]} D'_j$ is a dictator in the ℓth coordinate for ℓ. We would like to let $D'_j = \mathbb{Z}_t$ in every coordinate except ℓ. If this is not the case, then

there exists $x \in \mathbb{Z}_t^L$ such that $x_\ell \in D'_\ell$, but $f(x)$ is not equal to the the common value of $f(y)$ where $y \in T'$ and $x_\ell = y_\ell$. Let T_x be the subset of $\mathbb{Z}_t^L$ disjoint from x. We claim that f restricted to T_x satisfies the $[k-1, t-1, 2]$ gadget. If there were a disjoint set of size $k-1$ within T_x which all take on the value $f(x)$, then adding in x, we get a disjoint set of size k which all take on the value $f(x)$, a contradiction. Now, consider the case when there is a disjoint set of size $k-1$ within T_x which all take on the value $1-f(x)$, Since $\lfloor 2t/3 \rfloor \geq k$, $2(t-k) > k-1$, and no point in this set take on the value x_ℓ in the ℓth coordinate, there exists $y \in T'$ such that $y_\ell = x_\ell$ which is disjoint from every element of this set. Since $f(y) = 1 - f(x)$, we have that there then is a disjoint of size k such that f takes on a constant value, yielding a contradiction. Thus, f restricted to T_x satisfies the $[k-1, t-1, 2]$ gadget. Since $k - 1 > \lceil (t-1)/2 \rceil$ and $k - 1 \leq \lfloor 2(t-1)/3 \rfloor$, we have by the induction hypothesis there is a grid of width at least $2((t-1)-(k-1))+2 = 2(t-k)+2$ which has the desired properties. Thus, we are done.

Hence, we may now assume that $D'_j = \mathbb{Z}_t$ for all $j \neq \ell$. Since $2(t-k) \geq 2$, we may select x^ℓ, y^ℓ such that $f(x^\ell) = 0, f(y^\ell) = 1$ and $x^\ell_\ell, y^\ell_\ell \in D'_\ell$. Repeating the same argument where we replace x^1, y^1 with x^ℓ, y^ℓ we may deduce that either we have the desired conclusion or there exists a set $D''_m \subset \mathbb{Z}_t$ of size $2(t-k)$ such that f is a dictator in the mth coordinate when the mth coordinate is in D''_m If $m \neq \ell$, then $f(x)$ must be a function of only x_m (respectively x_ℓ) when $(x_m, x_\ell) \in D''_m \times D'_\ell$. Since both dictators take on both the value 0 and 1, this is impossible. Thus, $\ell = m$. Let $D_\ell = D'_\ell \cup D''_\ell$. Since $x^\ell_\ell, y^\ell_\ell \notin D''_\ell$ we have that $|D_\ell| \geq 2(t-k)+2$, and there are at least $t-k+1$ values of D_ℓ which take on 0 with respect to f (respectively 1). For the other D_j's we can select an arbitrary $2(t-k)+2$ element subset of $\mathbb{Z}_t$. ◀

▶ **Corollary 3.12.** *If $k \leq \lceil 2t/3 \rceil$, then the choice of i is unique.*

Proof. For sake of contradiction, imagine that there are $i \neq i'$ and and families D_j and D'_j satisfying the properties of Lemma 3.10. For all $j \in [L]$, we have that $|D_j \cap D'_j| \geq |D_j| + |D'_j| - t = 3t - 4k + 4$. Note that f restricted to $\bigtimes_j (D_j \cap D'_j)$ is a dictator in i and a dictator in i'. Thus, f must be a constant function. Since the dictator is evenly split between 0s and 1s, we have that $|D_i \cap D'_i| \leq t - k + 1$. Thus, $3t - 4k + 4 \leq t - k + 1$ or $k \geq 2t/3 + 1$, which contradicts the assumed bound on k. ◀

Thus, if $k \leq \lfloor 2t/3 \rfloor$, a natural choice for $\mathsf{Dec}(f)$ is this unique i for which there is a large "sub-dictator." We now show that this is indeed a dictator.

▶ **Claim 3.13.** *Let $f : \mathbb{Z}_t^L \to \mathbb{Z}_2$ satisfy the $[k, t, 2]$ gadget for $k \leq \lfloor 2t/3 \rfloor$. Let $\mathsf{Dec}(f)$ be the unique $i \in [L]$ from Corollary 3.12. Then, Dec is a decoder.*

Proof. From Lemma 3.10 and Corollary 3.12, we have that Dec is nontrivial, sensible, and bounded. It remains to prove that Dec is compatible. Assume for sake of contradiction that there exists $f, g : \mathbb{Z}_t^L \to \mathbb{Z}_2$ which satisfy the $[k, t, 2]$ co-gadget, but $\{i\} = \mathsf{Dec}(f) \neq \{i'\} = \mathsf{Dec}(g)$. Let $D_j \subseteq \mathbb{Z}_t$ and $D'_j \subseteq \mathbb{Z}_t$ be the sets guaranteed by Lemma 3.10 for f and g, respectively. Additionally, let $T_0 \subset D_i$ for which f takes on the value 0. Note that $|T_0| = t - k + 1$, so $|D'_i \setminus T_0| \geq (2t - 2k + 2) - (t - k + 1) = t - k + 1$. Thus, there exists disjoint $A_g \subset \mathbb{Z}_t^L$ such that $|A_g| = t - k + 1$, and $g[A_g] = \{0\}$. Since $|A_g| = t - k + 1$ and $a_i \notin T_0$ for all $a_i \in A_g$, we have that we may select $A_f \subset \mathbb{Z}_t^L$ such that $|A_f| = t - k + 1$, $f[A_f] = \{0\}$, and $A_f \cup A_g$ is disjoint. Since f and g satisfy the $[k, t, 2]$ co-gadget, we have that $k - 1 \geq |A_f \cup A_g| = 2t - 2k + 2$. Thus, $k \geq 2t/3 + 1$, a contradiction. ◀

For this decoder to work well with our label cover reduction in Section 4, we need to show that this decoder is robust.

▶ **Lemma 3.14.** *Let $f : \mathbb{Z}_t^L \to \mathbb{Z}_2$ satisfy the $[k,t,2]$ gadget for $k \leq \lfloor 2t/3 \rfloor$. Then for all projections $\pi : [L] \to [L]$,*

$$\pi[\mathsf{Dec}(f)] = \mathsf{Dec}(f \square \pi).$$

That is, Dec *is robust.*

Proof. Assume for sake of contradiction that there exists π and $i \neq i'$ such that $\mathsf{Dec}(f) = \{i\}$, $\mathsf{Dec}(f\square\pi) = \{i'\}$ and $\pi(i) \neq i'$. Let $D_j \subseteq \mathbb{Z}_t$ be the family of 'dictator-bounding' sets of f guaranteed by Lemma 3.10, and likewise let $D'_j \subseteq \mathbb{Z}_t$ be the corresponding family of sets for $f\square\pi$. Let $T_0 \subset D_i$ be the subsets of values for which f takes on the value 0 in the Cartesian product of the D_ℓ's. Similarly, let $T'_0 \subset D'_j$ be the subset of values for which $f\square\pi$ takes on the value 0 in the Cartesian product of the D'_ℓ's.

Note that $|T_0| = |T'_0| = t-k+1$. Thus, $|D'_{\pi(i)} \setminus T_0| \geq 2t - 2k + 2 - (t-k+1) = t-k+1$. Therefore, we may select a disjoint set $S' \subseteq \mathbb{Z}_t^L$ such that $|S'| = t-k+1$, $f_\pi(s) = 0$, and $s_i \in D'_{\pi(i)} \setminus T_0$ for all $s \in S'$. Let $S_0 = \{s \in \mathbb{Z}_t^L : \text{exists } t \in S' \text{ such that } s_j = t_{\pi(j)} \text{ for all } j \in [L]\}$. Note that S_0 is also disjoint and $|S_0| = |S'|$. Since for all $s \in S_0$, $s_i \notin T_0$, we may select a disjoint set S_1 of size $t-k+1$ disjoint from S_0 such that $f(s) = 0$ for all $s \in S_1$. Thus, $S_0 \cup S_1$ is a disjoint set of size $2t-2k+2$ which takes on only 0s as values. Since f satisfies the $[k,t,2]$ gadget, $2t-2k+2 \leq k-1$. Thus, $k \geq 2t/3+1$, a contradiction. ◀

4 Hardness of Gadget Decoders

4.1 The projection co-gadget

In our label cover reduction in this section, we need to be able to integrate projections in our co-gadget constraints. To do that, we generalize the co-gadget to deal with arbitrary projections. The co-gadget constraints are similar to the edge constraints in [3].

▶ **Definition 4.1.** Let $k,t,c,\ell \geq 2$, $L \geq 1$ be positive integers such that $k \leq t$, $c \geq t/(k-1)$. Let $f, g : \mathbb{Z}_t^L \to \mathbb{Z}_c$ be a functions and $\pi : [L] \to [L]$ be a projection. We say that (f,g) satisfy the $[k,t,c]$ *π-co-gadget* if for all $A \subset \mathbb{Z}_t^L$ disjoint with $|A| = k$ and all partitions $A_1 \cup A_2 = A$ such that

- (Strong constraint) for all $x \in A_1$, $y \in A_2$, $a \in [L]$, then $x_a \neq y_{\pi(a)}$

we have that

$$|\{f(x) : x \in A_1\} \cup \{g(y) : y \in A_2\}| \geq 2.$$

From the definition, it is clear that the π-co-gadget constraint can be implemented as a k-uniform t-strong hypergraph. The following claim is the main motivation for the previous definition.

▶ **Lemma 4.2.** *Let k,t,c,L be positive integers such that $k \leq t$, $c \geq t/(k-1)$. Let $f, g : \mathbb{Z}_t^L \to \mathbb{Z}_c$ be functions and $\pi : [L] \to [L]$ a projection. If (f,g) satisfy the $[k,t,c]$ π-co-gadget, then $(f\square\pi, g)$ satisfy the $[k,t,c]$ co-gadget.*

▶ Remark. It turns out the converse is false: the π-co-gadget between f and g is strictly stronger than the co-gadget on the projections. In spite of this, the reduction in power is offset by the modularity achieved by having the robust decoder act as a liaison between the coloring gadget and the label cover reduction.

Proof. It suffices to show that every $[k,t,c]$ co-gadget constraint on $(f\square\pi, g)$ is reflected in a constraint in π-co-gadget for (f,g).

Consider every disjoint $A \subseteq \mathbb{Z}_t^L$ such that $|A| = k$. Let $A_1 \cup A_2 = A$ be a partition of A. We seek to show that

$$|\{(f\square\pi)(x) : x \in A_1\} \cup \{g(y) : y \in A_2\}| \geq 2.$$

Define $B_1 \subseteq \mathbb{Z}_t^L$ to be

$$B_1 = \{z \in \mathbb{Z}_t^L : \text{there exists } x \in A_1 \text{ such that } z_j = x_{\pi(j)} \text{ for all } j \in [L]\}.$$

For each $x \in A_1$, there is a unique corresponding $z \in B_1$, and vice versa, so $|A_1| = |B_1|$ We claim that (B_1, A_2) is a hyperedge constraint in Definition 4.1. Clearly $|B_1| + |A_2| = k$ since $|A_1| = |B_1|$. For any $z \in B_1, y \in A_2$, we have that there is a $x \in A_1$ such that $z_j = x_{\pi(j)}$ for all $j \in [L]$. Since (A_1, A_2) is a valid hyperedge for the co-gadget, $x_{\pi(j)} \neq y_{\pi(j)}$ for all $j \in [L]$. Thus, $z_j \neq y_{\pi(j)}$ for all $j \in [L]$. Therefore, (B_1, A_2) is a valid hyperedge in the π-co-gadget so

$$|\{f(z) : z \in B_1\} \cup \{g(y) : y \in A_2\}| \geq 2.$$

Now, by definition of B_1, we have that $\{f(z) : z \in B_1\} = \{(f\square\pi)(x) : x \in A_1\}$. Thus,

$$|\{(f\square\pi)(x) : x \in A_1\} \cup \{g(y) : y \in A_2\}| \geq 2.$$

Thus, $(f\square\pi, g)$ satisfy this constraint in the co-gadget. Since the choice of this constraint was arbitrary, we have that $(f\square\pi, g)$ satisfy the $[k,t,c]$ co-gadget, as desired. ◀

4.2 The main reduction

Like [3], to obtain NP-hardness results, we reduce from *Label Cover*.

▶ **Definition 4.3.** An instance of *Label Cover* consists of $\Psi = (U, V, E, [L], \{\pi_e : [L] \to [L]\}_{e \in E})$ a bipartite graph for which each edge has been assigned a projection constraint. The constraint satisfaction problem is to find labelings $\sigma_1 : U \to [L], \sigma_2 : V \to [L]$ of the vertices such that for all $(u,v) \in E$, $\pi_{(u,v)}(\sigma_1(u)) = \sigma_2(v)$.

Although label cover is well-known to be hard with a large approximation gap, we only need that the problem of finding a fully correct labeling is NP-hard.

▶ **Lemma 4.4.** *It is* NP*-hard to determine if a Label Cover instance* Ψ *is satisfiable (whether a correct labeling exists), where* L *is a constant.*

Proof for completeness. We show that we can take $L = 6$ by reducing from 3-coloring. Let $G = (V, E)$ be a graph we seek to three color. We let the U of our Ψ be the set of edges E and we let V of our Ψ be the vertices V. If our color set is $\{0,1,2\}$, we identify our label set with $\{(0,1),(0,2),(1,0),(1,2),(2,0),(2,1)\}$, the six possible colorings of an edge $(u,v) \in E$. We also identify our label set with $\{0,1,2,3,4,5\}$. For $(u,v) \in E$, we have edge from (u,v) to u and one from (u,v) to v. The projection constraints are

$$\pi_{(u,v)\to u}((c_1,c_2)) = c_1, \pi_{(u,v)\to v}((c_1,c_2)) = c_2$$

Note that we do not use labels 3, 4, and 5 on the right side of Ψ. Now, it is easy to check given these constraints that if G has a three coloring $\gamma : V \to \{0,1,2\}$, then the labelings for all $(u,v) \in E$ and $u \in V$

$$\sigma_1((u,v)) = (\gamma(u), \gamma(v)), \sigma_2(u) = \gamma(u)$$

will satisfy Ψ. Conversely, any correct labeling for Ψ will correspond to a proper coloring from G. Thus, since 3-coloring is NP-complete, Label Cover is NP-hard, as desired. ◀

▶ **Lemma 4.5.** *If the $[k,t,c]$-coloring gadget ($k \le t$) has a robust decoder $\mathsf{Dec} = \mathsf{Dec}_{[k,t,c]}$ such that Dec always decodes into a unique coordinate ($C(k,t,c) = 1$), then $[k,t,c]$-coloring is NP-hard.*

Proof. We reduce from label cover. Let $\Psi = (U, V, E, [L], \{\pi_e : [L] \to [L]\}_{e \in E})$ be our instance of label cover. Replace each vertex $u \in U$ and $v \in V$ with $[k,t,c]$-gadgets whose colorings are represented by f_u and f_v, respectively. Replace each $(u,v) \in E$ with projection $\pi_{(u,v)}$ with the $\pi_{(u,v)}$-co-gadget for f_u and f_v. Call the resulting (hyper)graph G_Ψ. Since L is a constant G_Ψ is polynomial (in fact, linear) in the size of Ψ. To complete the reduction it suffices to prove the following

▶ **Claim 4.6** (Completeness). *If Ψ is satisfiable, then G_Ψ is t-strong colorable.*

Proof. Let $\sigma_1 : U \to [L]$ and $\sigma_2 : V \to [L]$ be the labelings. For all $u \in U$ and $v \in V$ and $x \in \mathbb{Z}_t^L$ let

$$f_u(x) = x_{\sigma_1(u)}, f_v(x) = x_{\sigma_2(v)}.$$

Clearly this is a t-coloring of G_Ψ. Now, we show that every hyperedge is t-strong colored. In every $[k,t,c]$-gadget every two vertices in each hyperedge differ in every coordinate, so their colors must be different. For any $(u,v) \in E$, if not every $[k,t,c]$ $\pi_{(u,v)}$-co-gadget constraint is t-strongly colored, then there are $x, y \in \mathbb{Z}_t^L$ such that $f_u(x) = f_v(y)$ but $x_j \neq y_{\pi_{(u,v)}(j)}$ for all j (we cannot possible have two vertices of the same color in the same hyperedge on the same side of the bipartite graph). In particular, this implies that $x_{\sigma_1(u)} = f_u(x) = f_v(y) = y_{\sigma_2(v)} = y_{\pi_{(u,v)}(\sigma_1(u))}$, contradicting our assumption about the labeling. Thus, G_Ψ is indeed t-strong colorable. ◀

▶ **Claim 4.7** (Soundness). *If G_Ψ is c-colorable, then Ψ is satisfiable.*

Proof. From the assumption we know there exist $\{f_u : u \in U\}$ and $\{f_v : v \in V\}$ which satisfy the $[k,t,c]$ gadget. Thus, since Dec does unique decoding, we may set

$$\sigma_1(u) = \mathsf{Dec}(f_u), \sigma_2(v) = \mathsf{Dec}(f_v).$$

It suffices to check for all $(u,v) \in E$ that $\pi_{(u,v)}(\sigma_1(u)) = \sigma_2(v)$. Since f_u and f_v satisfy the $[k,t,c]$ $\pi_{(u,v)}$-co-gadget (by construction), we have that $f_u \square \pi_{(u,v)}$ and f_v satisfy the $[k,t,c]$-co-gadget. Thus, since Dec is a decoder with unique decoding, $\mathsf{Dec}(f_u \square \pi_{(u,v)}) = \mathsf{Dec}(f_v)$. Because Dec is robust, $\pi_{(u,v)}(\mathsf{Dec}(f_u)) = \mathsf{Dec}(f_u \square \pi_{(u,v)})$. Thus, $\pi_{(u,v)}(\sigma_1(u)) = \sigma_2(v)$ for all $(u,v) \in E$, so Ψ is satisfiable, as desired. ◀

Thus, we have reduced from Label Cover to $[k,t,c]$-coloring, so $[k,t,c]$-coloring is NP-hard. ◀

From this, Theorem 1.1 follows from Lemma 3.9. Theorem 1.3 follows from Claim 3.3, Lemma 3.14, and Lemma 1.6.

▶ Remark. Using the techniques of [3], we can drop the constraint that $C(k,t,c) = 1$ by reducing from Label Cover with an approximation gap.

▶ Remark. Label Cover is also NP-hard if Ψ has bounded degree. This follows from the hardness of 3-coloring on bounded-degree graphs and applying the reduction in Lemma 4.4. Our reduction then shows that *bounded degree* $[k,t,c]$-coloring is NP-hard, obtaining a result similar to that of [13].

Acknowledgments. We thank the anonymous referees for their helpful comments.

References

1. N. Alon, I. Dinur, E. Friedgut, and B. Sudakov. Graph products, fourier analysis and spectral techniques. *Geometric & Functional Analysis GAFA*, 14(5):913–940, 2004. doi:10.1007/s00039-004-0478-3.
2. Noga Alon. personal communication, Oct 2014.
3. Per Austrin, Venkatesan Guruswami, and Johan Håstad. (2 + epsilon)-SAT is NP-hard. In *Proceedings of the 55th IEEE Symposium on Foundations of Computer Science*, pages 1–10, Oct 2014.
4. Andrei A. Bulatov, Peter Jeavons, and Andrei A. Krokhin. Classifying the complexity of constraints using finite algebras. *SIAM J. Comput.*, 34(3):720–742, 2005.
5. Irit Dinur and Venkatesan Guruswami. PCPs via low-degree long code and hardness for constrained hypergraph coloring. In *Proceedings of the 54th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 340–349, October 2013.
6. Irit Dinur, Elchanan Mossel, and Oded Regev. Conditional hardness for approximate coloring. *SIAM Journal on Computing*, 39(3):843–873, 2009. doi:10.1137/07068062X.
7. Irit Dinur, Oded Regev, and Clifford Smyth. The hardness of 3-uniform hypergraph coloring. *Combinatorica*, 25(5):519–535, September 2005. doi:10.1007/s00493-005-0032-4.
8. Irit Dinur and Igor Shinkar. On the conditional hardness of coloring a 4-colorable graph with super-constant number of colors. In Maria Serna, Ronen Shaltiel, Klaus Jansen, and José Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, volume 6302 of *Lecture Notes in Computer Science*, pages 138–151. Springer Berlin Heidelberg, 2010. doi:10.1007/978-3-642-15369-3_11.
9. M. R. Garey and David S. Johnson. The complexity of near-optimal graph coloring. *Journal of the ACM*, 23(1):43–49, 1976.
10. D. Greenwell and L. Lovász. Applications of product colouring. *Acta Mathematica Academiae Scientiarum Hungarica*, 25(3-4):335–340, 1974. doi:10.1007/BF01886093.
11. Venkatesan Guruswami, Prahladh Harsha, Johan Håstad, Srikanth Srinivasan, and Girish Varma. Super-polylogarithmic hypergraph coloring hardness via low-degree long codes. In *Proceedings of the 46th ACM Symposium on Theory of Computing (STOC)*, pages 614–623, 2014.
12. Venkatesan Guruswami, Johan Håstad, and Madhu Sudan. Hardness of approximate hypergraph coloring. *SIAM Journal on Computing*, 31(6):1663–1686, 2002.
13. Venkatesan Guruswami and Sanjeev Khanna. On the hardness of 4-coloring a 3-colorable graph. *SIAM J. Discrete Math.*, 18(1):30–40, 2004.
14. Venkatesan Guruswami and Euiwoong Lee. Strong inapproximability results on balanced rainbow-colorable hypergraphs. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, pages 822–836, 2015.
15. Venkatesan Guruswami, Prasad Raghavendra, Rishi Saket, and Yi Wu. Bypassing ugc from some optimal geometric inapproximability results. In *Proceedings of the Twenty-third Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'12, pages 699–717. SIAM, 2012. URL: http://dl.acm.org/citation.cfm?id=2095116.2095174.
16. Johan Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, 2001.
17. Jonas Holmerin. Vertex cover on 4-regular hypergraphs is hard to approximate within 2-epsilon. In *Proceedings on 34th Annual ACM Symposium on Theory of Computing*, pages 544–552, 2002.
18. Sangxia Huang. Approximation resistance on satisfiable instances for predicates with few accepting inputs. In *Proceedings of the Forty-fifth Annual ACM Symposium on Theory of Computing*, STOC'13, pages 457–466, New York, NY, USA, 2013. ACM. doi:10.1145/2488608.2488666.

19 Sangxia Huang. Improved hardness of approximating chromatic number. In Prasad Raghavendra, Sofya Raskhodnikova, Klaus Jansen, and JoséD.P. Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, volume 8096 of *Lecture Notes in Computer Science*, pages 233–243. Springer Berlin Heidelberg, 2013. doi:10.1007/978-3-642-40328-6_17.

20 Sangxia Huang. $2^{(\log N)^{1/10-o(1)}}$ hardness for hypergraph coloring. *Electronic Colloquium on Computational Complexity (ECCC)*, 22:62, 2015. URL: http://eccc.hpi-web.de/report/2015/062.

21 Sanjeev Khanna, Nathan Linial, and Shmuel Safra. On the hardness of approximating the chromatic number. *Combinatorica*, 20(3):393–415, 2000.

22 S. Khot. Improved inapproximability results for maxclique, chromatic number and approximate graph coloring. In *Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on*, pages 600–609, Oct 2001. doi:10.1109/SFCS.2001.959936.

23 Subhash Khot. Hardness results for coloring 3-colorable 3-uniform hypergraphs. In *Proceedings of the 43rd Symposium on Foundations of Computer Science*, pages 23–32, 2002.

24 Subhash Khot and Rishi Saket. Hardness of coloring 2-colorable 12-uniform hypergraphs with ith $2^{(\log n)^{\Omega(1)}}$ colors. In *55th IEEE Annual Symposium on Foundations of Computer Science*, pages 206–215, 2014.

25 Colin McDiarmid. A random recolouring method for graphs and hypergraphs. *Combinatorics, Probability and Computing*, 2:363–365, 9 1993. doi:10.1017/S0963548300000730.

26 Saul Stahl. The multichromatic numbers of some kneser graphs. *Discrete Mathematics*, 185:287–291, 1998.

27 Girish Varma. A note on reducing uniformity in Khot-Saket hypergraph coloring hardness reductions. unpublished, 2014. arXiv:1408.0262.

A Combinatorial Gadget Classifications

A.1 Claim 3.1

Proof of Claim 3.1. The proof given is similar to the combinatorial proof in [1] (Claim 4.1). Let $I = \{(i, \ldots, i) \in \mathbb{Z}_t^L : i \in \mathbb{Z}_t\}$. Clearly I and all of its translates are disjoint, so for all $x \in \mathbb{Z}_t^L$, $|S \cap (x + I)| \leq \omega(S)$. Since $|I| = t$,

$$t|S| = \sum_{x \in \mathbb{Z}_t^L} |S \cap (x + I)| \leq |\mathbb{Z}_t^L| \omega(S)$$

which implies (1). Note that equality holds if and only if $|S \cap (x + I)| = \omega(S)$. In fact, I can be replaced by any set of t disjoint points in $\mathbb{Z}_t^L$. It is easy to then see that if S is a dictator, equality always holds.

Now, we show if equality holds, then S is a dictator. We present a proof using the Fourier Analysis techniques of [1]. Let $f : \mathbb{Z}_t^L \to \{1, -1\}$ be the indicator function for S in the sense that $f(x) = -1$ if and only if $x \in S$. Using the notation of [1], let $\hat{f} : \mathbb{Z}_t^L \to \mathbb{C}$ be f's Fourier transform, and consider the following function:

$$A(f)(x) = \frac{1}{(t-1)^L} \sum_{y \in (\mathbb{Z}_t \setminus \{0\})^L} f(x + y)$$

Due to the structure proven above, it is easy to see that for all $x \in \mathbb{Z}_t^L$,

$$\begin{aligned} A(f)(x) &= \frac{1}{t-1}\sum_{i=1}^{t-1} f(x+(i,\dots,i)) \\ &= -\frac{f(x)}{(t-1)} + \frac{1}{t-1}\sum_{i=0}^{t-1} f(x+(i,\dots,i)) \\ &= -\frac{f(x)}{(t-1)} + \frac{1}{t-1}(t-2\omega(S)) \end{aligned}$$

Thus,

$$\widehat{A(f)}(0,\dots,0) = \frac{t-2\omega(S)}{t-1} - \frac{\hat{f}(0,\dots,0)}{t-1}$$

and if $x \neq (0,\dots,0)$,

$$\widehat{A(f)}(x) = -\hat{f}(x)/(t-1).$$

Let $|x|$ be the number of nonzero coordinates of x. Claim 4.3 of [1] shows though that

$$\widehat{A(f)}(x) = \hat{f}(x)\left(\frac{-\omega(S)}{t-1}\right)^{|x|},$$

Combining these two, we must have that $\hat{f}(x) = 0$ unless $|x| \leq 1$. Thus, f is nonzero on only its first two levels which Lemma 2.3 of [1] implies that f is a dictator, as desired.

A.2 Hardness of [4,5,2]-coloring

▶ **Claim 1.1.** *Consider $f : \mathbb{Z}_3 \times \mathbb{Z}_4^{L-1} \to \{0,1\}$ such that for all $x,y,z \in \mathbb{Z}_3 \times \mathbb{Z}_4^{L-1}$ disjoint, $\{f(x),f(y),f(z)\} = \{0,1\}$. Then, either there exists $a \in \mathbb{Z}_3$ such that $f(x)$ is constant for all $x \in \{a\} \times \mathbb{Z}_4^{L-1}$ or f is a dictator in one of the coordinates $\{2,\dots,L\}$.*

Proof. If $L = 1$, the claim is obvious. This proof proceeds by casework. If f is constant on all of coordinates $\{2,\dots,L\}$, then we are done. Otherwise, without loss of generality, we are in one of the following two cases

1. $f(0\dots00) = f(0\dots01) = f(0\dots02) = 0, f(0\dots03) = 1$
2. $f(0\dots00) = f(0\dots01) = 0, f(0\dots02) = f(0\dots03) = 1$

Case 1. If $f(1\dots1) = 1$, then we must have that $f(2\dots2) = 0$ which is "equivalent" to $f(1\dots1) = 0$. Thus, we only need to consider the case $f(1\dots1) = 0$. Now we have that following series of implications

$$f(2\{2,3\}^{L-2}\{0,2,3\}) = 1. \tag{9}$$

$$f(1\{1,2,3\}^{L-2}\{0,1,2\}) = 0. \tag{10}$$

$$f(2\{1,2,3\}^{L-2}\{0,1,2,3\}) = 1. \tag{11}$$

If $f(1x) = 0$ for all $x \in \mathbb{Z}_4^{L-1}$, then we are done. Otherwise, there exists $x_0 \in \mathbb{Z}_4^{L-1}$ such that $f(1x_0) = 1$. Using (11), this implies that $f(0y) = 0$ for all $y \in \mathbb{Z}_4^{L-1}$ disjoint from x_0. Furthermore, by (10) we can conclude that $f(2z) = 1$ for all $z \in \mathbb{Z}_4^{L-1}$ where z is disjoint from y. Since for all $z \in \mathbb{Z}_4^{L-1}$, we can find $y_0 \in \mathbb{Z}_4^{L-1}$ such that y_0 is disjoint from both x_0 and z, we must have that $f(2z) = 1$ for all $z \in \mathbb{Z}_4^{L-1}$, as desired.

Case 2. First assume that $f(1\dots1)=1$, then we get that

$$f(2\{2,3\}^{L-2}\{0,2,3\})=0. \tag{12}$$
$$f(1\{1,2,3\}^{L-2}\{0,1,2,3\})=1. \tag{13}$$
$$f(2\{1,2,3\}^{L-2}\{0,1,2,3\})=0. \tag{14}$$

if $f(1x)=1$ for all $x\in\mathbb{Z}_4^{l-1}$, we are done, else $f(1x_0)=0$ for some $x_0\in\mathbb{Z}_4^{L-1}$. Applying the same reasoning as in Case 1, we reach the same conclusion.
Now, we may assume that for all $x\in\{1,2\}\times\{1,2,3\}^{L-2}\times\{0,1,2,3\}$, $f(x)=0$ if and only if $x_L\in\{0,1\}$. Otherwise, we fall into a case already covered by permuting the coordinates or output labels. If $f(x)$ is a dictator in the last coordinate, we are done. Else, we may assume without loss of generality that there is $x\in\mathbb{Z}_3\times\mathbb{Z}_4^{L-2}$ such that $f(x0)=1$. Thus, $f(y\{2,3\})=0$ for all $y\in\mathbb{Z}_3\times\mathbb{Z}_4^{L-2}$ disjoint from x. But we also know that there is some $y'\in\{1,2\}\times\{1,2,3\}^{L-2}$ disjoint from x such that $f(y'\{2,3\})=1$, a contradiction. Thus, we have exhausted all cases. ◀

▶ **Corollary 1.2.** *Consider $f:\mathbb{Z}_3^k\times\mathbb{Z}_4^{L-k}\to\{0,1\}$ such that for all $x,y,z\in\mathbb{Z}_3^k\times\mathbb{Z}_4^{L-k}$ disjoint, $\{f(x),f(y),f(z)\}=\{0,1\}$. Then, either there exists $a\in\mathbb{Z}_3^k$ such that $f(x)$ is constant for all $x\in\{a\}\times\mathbb{Z}_4^{L-k}$ or f is a dictator in one of the coordinates $\{k+1,\dots,L\}$.*

Proof. For any three $x_0,y_0,z_0\in\mathbb{Z}_3^k$ disjoint, construct the map $f'_{x_0,y_0,z_0}:\mathbb{Z}_3\times\mathbb{Z}_4^{L-k}$, such that for all $w\in\mathbb{Z}_4^{L-k}$,

$$f'_{x_0,y_0,z_0}(\{0\}w)=f(x_0w)$$
$$f'_{x_0,y_0,z_0}(\{1\}w)=f(y_0w)$$
$$f'_{x_0,y_0,z_0}(\{2\}w)=f(z_0w).$$

It is clear that f'_{x_0,y_0,z_0} meets the hypothesis of Claim 1.1. Thus, either f'_{x_0,y_0,z_0} is a dictator in of coordinates $\{2,\dots,L-k+1\}$, or for one of $w_0\in\{x_0,y_0,z_0\}$, $f(w_0\times\mathbb{Z}_4^{L-K})$ is constant. Since the latter is sufficient to establish the claim, we assume the former in all cases. That is, for all disjoint $\{x_0,y_0,z_0\}\subseteq\mathbb{Z}_3^k$, f'_{x_0,y_0,z_0} is a dictator. Notice that this implies for all $x_0\in\mathbb{Z}_3^k$ there is $i_{x_0}\in\{k+1,\dots,L\}$ such that f restricted to $\{x_0\}\times\mathbb{Z}_4^{L-k}$. Additionally, note that for all $x_0,y_0\in\mathbb{Z}_3^k$ disjoint we must have that $i_{x_0}=i_{y_0}$ since there exists $z_0\in\mathbb{Z}_3^k$ disjoint from both x_0 and y_0. Since the "disjoint" property induces a connected graph on $\mathbb{Z}_3^k$, we have that i_{x_0} is constant for all $x_0\in\mathbb{Z}_3^k$. Thus, f is a dictator on one of $\{k+1,\dots L\}$, as desired. ◀

▶ **Lemma 1.3.** *For all $f:\mathbb{Z}_5^L\to\mathbb{Z}_2$ satisfying the $[4,5,2]$ gadget, there exists $i\in[L]$ and $a,b\in\mathbb{Z}_5$ distinct such that for all $x\in\{x\in\mathbb{Z}_5^L:x_i=a \text{ or } x_i=b\}$, $f(x)$ is constant.*

Proof. First, we show (up to symmetry) a wide class of f have this property.

▶ **Claim 1.4.** *Let $f:\mathbb{Z}_5^L\to\mathbb{Z}_2$ satisfy the $[4,5,2]$ gadget and assume that f restricted to $\mathbb{Z}_4^L$ satisfies the $[3,4,2]$ gadget. Then f satisfies the conclusion of Lemma 1.3.*

Proof. Clearly by Claim 3.2 f restricted to $\mathbb{Z}_4^L$ is a dictator in one of the coordinates. Assume without loss of generality that $f(\{0,1\}\times\mathbb{Z}_4^{L-1})=0$ and $f(\{2,3\}\times\mathbb{Z}_4^{L-1})=1$. In order for the claim to not be immediately true, we must have without loss of generality that there exist $x,y\in\mathbb{Z}_4^{L-1}$ such that $f(0x)=1$ and $f(2y)=0$. Thus, $f(4z)=0$ for all $z\in\mathbb{Z}_5^{L-1}$ disjoint from x and $f(4z)=1$ for all $z\in\mathbb{Z}_5^{L-1}$ disjoint from y. Since there are $z\in\mathbb{Z}_5^{L-1}$ disjoint from both x and y, we have a contradiction. Thus, the claim is true. ◀

Now, we show an even wider class of f satisfy the lemma.

▶ **Claim 1.5.** *Let $f : \mathbb{Z}_5^L \to \mathbb{Z}_2$ satisfy the $[4,5,2]$ gadget and assume that f restricted to $\{0\} \times \mathbb{Z}_4^{L-1}$ is always 0. Then either f restricted to $\{0\} \times \mathbb{Z}_5^{L-1}$ or f satisfies the conclusion of Lemma 1.3.*

Proof. Assume the first conclusion is false. Thus, there is $x \in \mathbb{Z}_5^{L-1}$ such that $f(0x) = 1$. Consider the hypercube H_{0x} of points disjoint from $0x$. If any three disjoint points in H_{0x} have the same value, we could find a fourth value in $\mathbb{Z}_5^{L-1}$ which is disjoint from all three but has the same value, a contradiction. Thus, f restricted to H_{0x} satisfies the $[3,4,2]$ gadget. Thus, by Claim 1.4 we have that f satisfies the conclusion of Lemma 1.3. ◀

Now, we may finish the proof. Clearly f must depend in at least one coordinate. Assume that $f(0\ldots0) = 0$ and $f(10\ldots0) = 1$. Thus, f restricted to $S = \{2,3,4\} \times \{1,2,3,4\}^{L-1}$ meets the hypothesis of Claim 1.1. Therefore, either f restricted to this set is a dictator or f has be constant on $\{a\} \times \{1,2,3,4\}^{L-1}$ for some $a \in \{2,3,4\}$. First, assume that the former case occurs and that f is dictated by coordinate $i \in \{2,\ldots,L\}$. Now, assume without loss of generality that $f(20\ldots0) = 1$. Thus, f restricted to $T = \{1,3,4\} \times \{1,2,3,4\}^{L-1}$ also meets the hypothesis of 1.1. Because S and T have a large overlap, it is not possible for f restricted to T to T to be dictated by any coordinate other than i. But it is possible for $f(\{1\} \times \{1,2,3,4\}^{L-1})$ to be constant. In the first case, f restricted to $\{1,2,3,4\}^L$ is also a dictator. Thus, f restricted to this set satisfies the $[3,4,2]$ gadget. Thus, by Claim 1.4, f satisfies the conclusion. In the second case, by Claim 1.5 f $f(\{1\} \times \mathbb{Z}_5^{L-1}) = 1$ or else we are done. This yields a contradiction since we can pick $x \in \{1\} \times \mathbb{Z}_5^{L-1}$ and $y, z \in \{3,4\} \times \{1,2,3,4\}^{L-1}$ such that x, y, z, and $20\ldots0$ are all disjoint but $f(x) = f(y) = f(z) = f(20\ldots0) = 1$ since f restricted to $\{3,4\} \times \{1,2,3,4\}^{L-1}$ is a dictator in a coordinate other than than the first.

Thus, we may now assume without loss of generality that f restricted to $\{2\} \times \{1,2,3,4\}^{L-1}$ is always 1. By 1.5, we may assume that $f(\{2\} \times \mathbb{Z}_5^{L-1}$ is always 1 (or else we are immediately done). Since $f(0\ldots0) = 0 \neq f(20\ldots0) = 1$, we have that f restricted to $\{1,3,4\} \times \{1,2,3,4\}^{L-1}$ also satisfies the $[3,4,2]$ gadget. Thus, $f(\{a\} \times \{1,2,3,4\}^{L-1}$ is constant and so we may assume that $f(\{a\} \times \mathbb{Z}_5^{L-1})$ is constant. Clearly if this constant value is 1 we are done, otherwise Thus, assume that $f(\{a\} \times \mathbb{Z}_5^{L-1}) = 0$. Therefore, $f(a0\ldots0) = 0$. Thus, f restricted to $(\mathbb{Z}_5 \setminus \{2,a\}) \times \mathbb{Z}_4^{L-1}$ also satisfies the $[3,4,2]$ gadget. Thus, there is $b \in \mathbb{Z}_5 \setminus \{2,a\}$ such that $f(b \times \{1,\ldots,4\}^{L-1})$ is constant, so $f(b \times \mathbb{Z}_5^{L-1})$ is constant (or else we are done). Thus, $i = 1$ and either $\{a,b\}$ or $\{1,b\}$ is the desired pair, as desired. ◀

▶ **Lemma 1.6.** *The $[4,5,2]$-coloring gadget has a robust decoder.*

Proof. We omit the proof. The proof is similar to that of Claim 3.13 and Lemma 3.14. ◀

A.3 Classification of $\langle t-1, t, 2\rangle$

In this subsection, we examine a balanced variant of the strong hypergraph coloring problem.

▶ **Definition 1.7.** Let $k, t, c \geq 2$ be positive integers with $t \geq k$. Define $\langle k,t,c\rangle$-*coloring* to be the following promise problem. Let G be a k-uniform hypergraph which is promised to be t-strong colorable. Can G be efficiently colored with c colors such that the discrepancy is minimal?

▶ **Definition 1.8.** Let L, k, t, c be positive integers with $t \geq k$, and let $f : \mathbb{Z}_t^L \to \mathbb{Z}_c$ be a function. We say that f *satisfies the* $\langle k, t, c\rangle$ *gadget* if for all $S \subset \mathbb{Z}_t^L$ such that $|S| = k$ and S is disjoint otherwise, we have that the multiset $\{f(x) : x \in S\}$ is as equi-distributed as possible.

Note the $\langle 3, 4, 2\rangle$ is equivalent to $[3, 4, 2]$ (both gadget and problem) and that $\langle 4, 5, 2\rangle$ is equivalent to $[4, 5, 2]$. We now prove a result that holds for $\langle t-1, t, 2\rangle$ for all odd t.

▶ **Lemma 1.9.** *If $f : \mathbb{Z}_t^L \to \mathbb{Z}_2$ satisfies the $\langle t-1, t, 2\rangle$ gadget, where t is even, then f is a dictator.*

Proof. We present a proof using the Fourier Analysis techniques of [1]. Using their notation, remap f so that its output if $\{-1, 1\}$ instead of $\{0, 1\}$, let $\hat{f} : \mathbb{Z}_t^L \to \mathbb{C}$ be f's Fourier transform, and consider the following function:

$$A(f)(x) = \frac{1}{(t-1)^L} \sum_{y \in (\mathbb{Z}_t \setminus \{0\})^L} f(x+y)$$

For combinatorial reasons, it is easy to see in our context that $A(f)(x) = -f(x)/(t-1)$ for all $x \in \mathbb{Z}_t^L$. Thus, $\widehat{A(f)}(x) = -\hat{f}(x)/(t-1)$. Claim 4.3 of their paper shows though that

$$\widehat{A(f)}(x) = \hat{f}(x) \left(\frac{-1}{t-1}\right)^{|x|}.$$

Combining these two, we must have that $\hat{f}(x) = 0$ unless $|x| = 1$. That is, x has only one nonzero coordinate. Thus, f is nonzero on only its first two levels which Lemma 2.3 of their paper implies that f is a dictator, as desired. ◀

We omit the proof that there exists a robust decoder and the subsequent Label Cover argument.

▶ **Definition 1.10.** A function $f : \mathbb{Z}_t^L \to \mathbb{Z}_2$ is an almost dictator if there exists an independent set I of $\mathbb{Z}_t^L$ (i.e., a subset no two of whose elements are disjoint) such that f restricted to $\mathbb{Z}_t^L \setminus I$ is a dictator.

▶ **Conjecture 1.11.** *If $f : \mathbb{Z}_t^L \to \mathbb{Z}_2$ satisfies the $\langle t-1, t, 2\rangle$ gadget, where t is odd, then f is an almost dictator.*

This conjecture, with a suitable application of Label Cover, would imply that finding a discrepancy two 2-coloring of a t-colorable graph is NP-hard. ◀

B Nonexistence of Robust Decoding of [2, 3, 6]

▶ **Claim 2.1.** *There does not exist a robust decoding of the $[2, 3, 6]$-coloring gadget. Even if the projections considered are $p(L)$-to-1 for any $p(L) = \omega(1)$.*

Proof. Assume for sake of contradiction that there exist a robust decoding Dec. Let $p(L) : \mathbb{N} \to \mathbb{N}$ be any function in $\omega(1)$. For all $L \geq 1$, consider $f : \mathbb{Z}_3^L \to \mathbb{Z}_6$ with the following properties.

- For any $x \in \mathbb{Z}_3^L$ such that there is $s \in \mathbb{Z}_3$ such that $|\{i \in [p(L)] : x_i = s\}| > p(L)/2$, then $f(x) = s + 3$.
- Otherwise, $f(x) = x_1$.

As a sanity check, note that for each $s \in \mathbb{Z}_6$, $f^{-1}(s)$ is an independent set. For each $S \subseteq [p(L)]$ such that $|S| > p(L)/2$, let π_S be the projection such that $\pi_S(i) = \min S$ if $i \in S$ and $\pi_S(i) = i$ otherwise. Since $|S| > p(L)/2$, $f \square \pi_S$ has a range of $\{3, 4, 5\}$. Furthermore, $f \square \pi_S$ is a dictator in coordinate $\min S$, so $\mathsf{Dec}(f \square \pi_S) = \{\min S\}$. Since Dec is robust, $(\pi_S)^{-1}(\min S) = S$ must have nontrivial intersection with $\mathsf{Dec}(f)$. Thus, $\mathsf{Dec}(f)$ must have nontrivial intersection with every S such that $|S| > p(L)/2$. Thus, $|\mathsf{Dec}(f)| \geq p(L)/2 = \omega(1)$ (since otherwise we could exhibit a non-intersecting S), but $|\mathsf{Dec}(f)| \leq C$ for some constant C independent of L, a contradiction. ◀

Note that this arguments suggests that the 'robust decoder' techniques could not work, unless we use a d-to-1 variant of label cover, of which hardness is only conjectured. A similar argument shows that there does not exist a robust decoding of the $[2, t, 2t]$-coloring gadget.

Invariance Principle on the Slice

Yuval Filmus[1], Guy Kindler[2], Elchanan Mossel[3], and Karl Wimmer[4]

1 **Technion – Israel Institute of Technology, Haifa, Israel**
yuvalfi@cs.technion.ac.il

2 **The Hebrew University of Jerusalem, Jerusalem, Israel**
gkindler@cs.huji.ac.il

3 **The Wharton School, University of Pennsylvania, Philadelphia, USA; and University of California, Berkeley, USA**
mossel@wharton.upenn.edu

4 **Duquesne University, Pittsburgh, USA**
wimmerk@duq.edu

Abstract

The non-linear invariance principle of Mossel, O'Donnell and Oleszkiewicz establishes that if $f(x_1, \ldots, x_n)$ is a multilinear low-degree polynomial with low influences then the distribution of $f(\mathcal{B}_1, \ldots, \mathcal{B}_n)$ is close (in various senses) to the distribution of $f(\mathcal{G}_1, \ldots, \mathcal{G}_n)$, where $\mathcal{B}_i \in_R \{-1, 1\}$ are independent Bernoulli random variables and $\mathcal{G}_i \sim \mathrm{N}(0, 1)$ are independent standard Gaussians. The invariance principle has seen many application in theoretical computer science, including the *Majority is Stablest* conjecture, which shows that the Goemans–Williamson algorithm for MAX-CUT is optimal under the Unique Games Conjecture.

More generally, MOO's invariance principle works for any two vectors of hypercontractive random variables $(\mathcal{X}_1, \ldots, \mathcal{X}_n), (\mathcal{Y}_1, \ldots, \mathcal{Y}_n)$ such that (i) *Matching moments*: $\mathcal{X}_i$ and $\mathcal{Y}_i$ have matching first and second moments, (ii) *Independence*: the variables $\mathcal{X}_1, \ldots, \mathcal{X}_n$ are independent, as are $\mathcal{Y}_1, \ldots, \mathcal{Y}_n$.

The independence condition is crucial to the proof of the theorem, yet in some cases we would like to use distributions $(\mathcal{X}_1, \ldots, \mathcal{X}_n)$ in which the individual coordinates are not independent. A common example is the uniform distribution on the *slice* $\binom{[n]}{k}$ which consists of all vectors $(x_1, \ldots, x_n) \in \{0, 1\}^n$ with Hamming weight k. The slice shows up in theoretical computer science (hardness amplification, direct sum testing), extremal combinatorics (Erdős–Ko–Rado theorems) and coding theory (in the guise of the Johnson association scheme).

Our main result is an invariance principle in which $(\mathcal{X}_1, \ldots, \mathcal{X}_n)$ is the uniform distribution on a slice $\binom{[n]}{pn}$ and $(\mathcal{Y}_1, \ldots, \mathcal{Y}_n)$ consists either of n independent $\mathrm{Ber}(p)$ random variables, or of n independent $\mathrm{N}(p, p(1-p))$ random variables. As applications, we prove a version of *Majority is Stablest* for functions on the slice, a version of Bourgain's tail theorem, a version of the Kindler–Safra structural theorem, and a stability version of the t-intersecting Erdős–Ko–Rado theorem, combining techniques of Wilson and Friedgut.

Our proof relies on a combination of ideas from analysis and probability, algebra and combinatorics. In particular, we make essential use of recent work of the first author which describes an explicit Fourier basis for the slice.

1998 ACM Subject Classification F.0 [Theory of Computation] General

Keywords and phrases analysis of boolean functions, invariance principle, Johnson association scheme, the slice

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.15

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 15; pp. 15:1–15:10

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

1 Introduction

Analysis of Boolean functions is an area at the intersection of theoretical computer science, functional analysis and probability theory, which traditionally studies Boolean functions on the Boolean cube $\{0,1\}^n$. A recent development in the area is the non-linear *invariance principle* of Mossel, O'Donnell and Oleszkiewicz [11], a vast generalization of the fundamental Berry–Esseen theorem. The Berry–Esseen theorem is a quantitative version of the Central Limit Theorem, giving bounds on the speed of convergence of a sum $\sum_i X_i$ to the corresponding Gaussian distribution. Convergence occurs as long as none of the summands X_i is too "prominent". The invariance principle is an analog of the Berry–Esseen theorem for low-degree polynomials. Given a low-degree polynomial f on n variables in which none of the variables is too prominent (technically, f has low *influences*), the invariance principle states that the distribution of $f(X_1,\ldots,X_n)$ and $f(Y_1,\ldots,Y_n)$ is similar as long as each of the vectors $(X_1,\ldots,X_n)$ and $(Y_1,\ldots,Y_n)$ consists of independent coordinates, the distributions of X_i, Y_i have matching first and second moments, and the variables X_i, Y_i are hypercontractive.

The invariance principle came up in the context of proving a conjecture, *Majority is Stablest*, claiming that the majority function is the most noise stable among functions which have low influences. It is often applied in the following setting: the X_i are skewed Bernoulli variables, and the Y_i are the matching normal distributions. The invariance principle allows us to analyze a function on the Boolean cube (corresponding to the X_i) by analyzing its counterpart in Gaussian space (corresponding to the Y_i), in which setting it can be analyzed using geometric methods. This approach has been used to prove many results in analysis of Boolean functions (see for example [8]).

The proof of the invariance principle relies on the product structure of the underlying probability spaces. The challenge of proving an invariance principle for non-product spaces seems far from trivial. Here we prove such an invariance principle for the distribution over $X_1,\ldots,X_n$ which is uniform over the *slice* $\binom{[n]}{k}$, defined as:

$$\binom{[n]}{k} = \{(x_1,\ldots,x_n) \in \{0,1\}^n : x_1 + \cdots + x_n = k\}.$$

This setting arises naturally in hardness of approximation, see e.g. [3], and in extremal combinatorics (the Erdős–Ko–Rado theorem and its many extensions).

Our invariance principle states that if f is a low-degree function on $\binom{[n]}{k}$ having low influences, then the distributions of $f(X_1,\ldots,X_n)$ and $f(Y_1,\ldots,Y_n)$ are close, where $X_1,\ldots,X_n$ is the uniform distribution on $\binom{[n]}{k}$, and $Y_1,\ldots,Y_n$ are either independent Bernoulli variables with expectation k/n, or independent Gaussians with the same mean and variance.

The classical invariance principle is stated only for low-influence functions. Indeed, high-influence functions like $f(x_1,\ldots,x_n) = x_1$ behave very differently on the Boolean cube and on Gaussian space. For the same reason, the condition of low-influence is necessary when comparing functions on the slice and on Gaussian space.

The invariance principle allows us to generalize two fundamental results to this setting: Majority is Stablest and Bourgain's tail bound. Using Bourgain's tail bound, we prove an analog of the Kindler–Safra theorem, which states that if a Boolean function is close to a function of constant degree, then it is close to a junta.

As a corollary of our Kindler–Safra theorem, we prove a stability version of the t-intersecting Erdős–Ko–Rado theorem, combining the method of Friedgut [7] with calculations of Wilson [12]. Friedgut showed that a t-intersecting family in $\binom{[n]}{k}$ of almost maximal size $(1-\epsilon)\binom{n-t}{k-t}$ is close to an optimal family (a t-star) as long as $\lambda < k/n < 1/(t+1) - \zeta$

(when $k/n > 1/(t+1)$, t-stars are no longer optimal). We extend his result to the regime $k/n \approx 1/(t+1)$.

The classical invariance principle is stated for *multilinear* polynomials, implicitly relying on the fact that every function on $\{0,1\}^n$ can be represented (uniquely) as a multilinear polynomial, and that multilinear polynomials have the same mean and variance under any product distribution in which the individual factors have the same mean and variance. In particular, the classical invariance principle shows that the correct way to lift a low-degree, low-influence function from $\{0,1\}^n$ to Gaussian space is via its multilinear representation.

The analogue of the collection of low degree multilinear functions on the discrete cube is given by the collection of low degree multilinear polynomials annihilated by the operator $\sum_{i=1}^n \frac{\partial}{\partial x_i}$. Dunkl [4, 5] showed that every function on the slice has a unique representation as a multilinear polynomial annihilated by the operator $\sum_{i=1}^n \frac{\partial}{\partial x_i}$. We call a polynomial satisfying this condition a *harmonic function*. In a recent paper [6], the first author showed that low-degree harmonic functions have *similar* mean and variance under both the uniform distribution on the slice and the corresponding Bernoulli and Gaussian product distributions. This is a necessary ingredient in our invariance principle.

Our results also apply for function on the slice that are not written in their harmonic representation. Starting with an arbitrary multilinear polynomial f, there is a unique harmonic function $\tilde{f}$ agreeing with f on a given slice. We show that as long as f depends on few coordinates, the two functions f and $\tilde{f}$ are close as functions over the Boolean cube. This implies that f behaves similarly on the slice, on the Boolean cube, and on Gaussian space.

Our proof combines algebraic, geometric and analytic ideas. A coupling argument, which crucially relies on properties of harmonic functions, shows that the distribution of a low-degree, low-influence harmonic function f is approximately invariant when we move from the original slice to nearby slices. Taken together, these slices form a thin layer around the original slice, on which f has roughly the same distribution as on the original slice. The classical invariance principle implies that the distribution of f on the layer is close to its distribution on the Gaussian counterpart of the layer, which turns out to be *identical* to its distribution on all of Gaussian space, completing the proof.

A special case of our main result can be stated as follows.

▶ **Theorem 1.1.** *For every $\epsilon > 0$ and integer $d \geq 0$ there exists $\tau = \tau(\epsilon, d) > 0$ such that the following holds. Let $n \geq 1/\tau$, and let f be a harmonic multilinear polynomial of degree d such that with respect to the uniform measure ν_{pn} on the slice $\binom{[n]}{pn}$, the variance of f is at most 1 and all influences of f are bounded by τ.*

The CDF distance between the distribution of f on the slice ν_{pn} and the distribution of f under the product measure μ_p with marginals $\mathrm{Ber}(p)$ *is at most ϵ: for all $\sigma \in \mathbb{R}$,*

$$|\Pr_{\nu_{pn}}[f < \sigma] - \Pr_{\mu_p}[f < \sigma]| < \epsilon.$$

Subsequent to this work, the first and third author came up with an alternative proof of Theorem 1.1 [10] which doesn't require the influences of f to be bounded. The proof is completely different, connecting the measures μ_p and ν_{pn} directly without recourse to Gaussian space. While the main result of [10] subsumes the main result of this paper, we believe that both approaches have merit. Furthermore, the applications of the invariance principle appearing here are not reproduced in [10].

Paper organization

An overview of our main results and methods appears in Section 2. Some open problems are described in Section 3. All proofs have been relegated to the full version of the paper, available online at `http://arxiv.org/abs/1504.01689`.

2 Overview

The goal of this section is to provide an overview of the results proved in this paper and the methods used to prove them. It is organized as follows. Some necessary basic definitions appear in Subsection 2.1. The invariance principle, its proof, and some standard consequences are described in Subsection 2.2. Some applications of the invariance principle appear in Subsection 2.3: versions of Majority is stablest, Bourgain's theorem, and the Kindler–Safra theorem for the slice. An application of the Kindler–Safra theorem to extremal combinatorics is described in Subsection 2.4. Finally, Subsection 2.5 presents results for non-harmonic multilinear polynomials.

2.1 Basic definitions

Measures

Our work involves three main probability measures, parametrized by an integer n and a probability $p \in (0,1)$:

- μ_p is the product distribution supported on the Boolean cube $\{0,1\}^n$ given by $\mu_p(S) = p^{|S|}(1-p)^{n-|S|}$.
- ν_{pn} is the uniform distribution on the slice $\binom{[n]}{pn} = \{(x_1,\ldots,x_n) \in \{0,1\}^n : x_1+\cdots+x_n = pn\}$ (we assume pn is an integer).
- $\mathcal{G}_p$ is the Gaussian product distribution $\mathrm{N}((p,\ldots,p), p(1-p)I_n)$ on Gaussian space $\mathbb{R}^n$.

We denote by $\|f\|_\pi$ the L2 norm of the polynomial f with respect to the measure π.

Harmonic polynomials

As stated in the introduction, we cannot expect an invariance principle to hold for all multilinear polynomials, since for example the polynomial $x_1+\cdots+x_n-pn$ vanishes on the slice but not on the Boolean cube or on Gaussian space. We therefore restrict our attention to *harmonic* multilinear polynomials, which are multilinear polynomials f satisfying the differential equation

$$\sum_{i=1}^{n} \frac{\partial f}{\partial x_i} = 0.$$

(The name *harmonic*, whose common meaning is different, was lifted from the literature.)

Dunkl [4, 5] showed that every function on the slice $\binom{[n]}{pn}$ has a unique representation as a harmonic multilinear polynomial whose degree is at most $\min(pn,(1-p)n)$. This is the analog of the well-known fact that every function on the Boolean cube has a unique representation as a multilinear polynomial.

One crucial property of low-degree harmonic multilinear polynomials is invariance of their L2 norm: for any $p \le 1/2$ and any harmonic multilinear polynomial f of degree $d \le pn$,

$$\|f\|_{\mu_p} = \|f\|_{\mathcal{G}_p} = \|f\|_{\nu_{pn}} \left(1 \pm O\left(\frac{d^2}{p(1-p)n}\right)\right).$$

This is proved in Filmus [6], and in fact this result (and its applications in the present work) was the main motivation for [6].

Influences

The classical definition of influence for a function f on the Boolean cube goes as follows. Define $f^{[i]}(x) = f(x^{[i]})$, where $x^{[i]}$ results from flipping the ith coordinate of x. The ith *cube*-influence of f is given by

$$\mathrm{Inf}_i^c[f] = \|f - f^{[i]}\|_{\mu_p}^2 = \left\| \frac{\partial f}{\partial x_i} \right\|_{\mu_p}^2 = \frac{1}{p(1-p)} \sum_{i \in S} \hat{f}(S)^2.$$

This notion doesn't make sense for functions on the slice, since the slice is not closed under flipping of a single coordinate. Instead, we consider what happens when two coordinates are swapped. Define $f^{(ij)}(x) = f(x^{(ij)})$, where $x^{(ij)}$ results from swapping the ith and jth coordinates of x. The (i,j)th *slice*-influence of f is given by

$$\mathrm{Inf}_{ij}^s[f] = \mathop{\mathbb{E}}_{\nu_{pn}}[(f - f^{(ij)})^2].$$

The influence of a single coordinate i is then defined as

$$\mathrm{Inf}_i^s[f] = \frac{1}{n} \sum_{j=1}^n \mathrm{Inf}_{ij}^s[f].$$

The two definitions are related: in the complete version of the paper we show that if $d = O(\sqrt{n})$ then

$$\mathrm{Inf}_i^s[f] = O_p\left(\frac{d}{n} \mathbb{V}[f] + \mathrm{Inf}_c^s[f] \right).$$

(The variance can be taken with respect to either the Boolean cube or the slice, due to the L2 invariance property.)

Noise stability

The classical definition of noise stability for a function f on the Boolean cube goes as follows:

$$\mathbb{S}_\rho^c[f] = \mathbb{E}[f(x)f(y)],$$

where $x \sim \mu_p$ and y is obtained from x by letting $y_i = x_i$ with probability ρ, and $y_i \sim \mu_p$ otherwise.

The analogous definition on the slice is slightly more complicated. For a function f on the slice,

$$\mathbb{S}_\rho^s[f] = \mathbb{E}[f(x)f(y)],$$

where $x \sim \nu_{pn}$ and y is obtained from x by doing $\mathrm{Po}(\frac{n-1}{2} \log \frac{1}{\rho})$ random transpositions (here $\mathrm{Po}(\lambda)$ is a Poisson distribution with mean λ). That this definition is the correct analog can be seen through the spectral lens:

$$\mathbb{S}_\rho^c[f] = \sum_d \rho^d \|f^{=d}\|_{\mu_p}^2, \qquad \mathbb{S}_\rho^s[f] = \sum_d \rho^{d - d(d-1)/n} \|f^{=d}\|_{\mu_{pn}}^2.$$

Here $f^{=d}$ is the dth homogeneous part of f consisting of all monomials of degree d.

2.2 Invariance principle

Our main theorem is an invariance principle for the slice.

▶ **Theorem 2.1.** *Let f be a harmonic multilinear polynomial of degree d such that with respect to ν_{pn}, $\mathbb{V}[f] \leq 1$ and $\mathrm{Inf}_i^s[f] \leq \tau$ for all $i \in [n]$. Suppose that $\tau \leq I_p^{-d}\delta^K$ and $n \geq I_p^d/\delta^K$, for some constants I_p, K. For any C-Lipschitz functional ψ and for $\pi \in \{\mathcal{G}_p, \mu_p\}$,*

$$|\mathop{\mathbb{E}}_{\nu_{pn}}[\psi(f)] - \mathop{\mathbb{E}}_{\pi}[\psi(f)]| = O_p(C\delta).$$

Proof sketch. Let ψ be a Lipschitz functional and f a harmonic multilinear polynomial of unit variance, low slice-influences, and low degree d. A simple argument (mentioned above) shows that f also has low cube-influences, and this implies that

$$\mathop{\mathbb{E}}_{\nu_k}[\psi(f)] \approx \mathop{\mathbb{E}}_{\nu_{pn}}[\psi(f)] \pm O_p\left(\frac{|k-np|}{\sqrt{n}} \cdot \sqrt{d}\right).$$

The idea is now to apply the multidimensional invariance principle jointly to f and to $S = \frac{x_1+\cdots+x_n-np}{\sqrt{p(1-p)n}}$, deducing

$$\mathop{\mathbb{E}}_{\mu_p}[\psi(f)\mathbf{1}_{|S|\leq\sigma}] = \mathop{\mathbb{E}}_{\mathcal{G}_p}[\psi(f)\mathbf{1}_{|S|\leq\sigma}] \pm \epsilon.$$

Let $\gamma_{p,q}$ be the restriction of $\mathcal{G}_p$ to the Gaussian slice $\{(x_1,\ldots,x_n) \in \mathbb{R}^n : x_1+\cdots+x_n = qn\}$. An easy argument shows that since f is harmonic, the distribution of $f(\mathcal{G}_p)$ and $f(\gamma_{p,q})$ is identical, and so

$$\mathop{\mathbb{E}}_{\mathcal{G}_p}[\psi(f)\mathbf{1}_{|S|\leq\sigma}] = \Pr_{\mathcal{G}_p}[|S| \leq \sigma] \mathop{\mathbb{E}}_{\mathcal{G}_p}[\psi(f)].$$

Similarly,

$$\mathop{\mathbb{E}}_{\mu_p}[\psi(f)\mathbf{1}_{|S|\leq\sigma}] = \Pr_{\mu_p}[|S| \leq \sigma](\mathop{\mathbb{E}}_{\mu_p}[\psi(f)] \pm O_p(\sigma\sqrt{d})).$$

Since $\Pr_{\mathcal{G}_p}[|S| \leq \sigma] \approx \Pr_{\mu_p}[|S| \leq \sigma] = \Theta_p(\sigma)$, we can conclude that

$$\mathop{\mathbb{E}}_{\nu_{pn}}[\psi(f)] \approx \mathop{\mathbb{E}}_{\mathcal{G}_p}[\psi(f)] \pm O_p\left(\sigma\sqrt{d} + \frac{\epsilon}{\sigma}\right).$$

By choosing σ appropriately, we balance the two errors and obtain our invariance principle. ◀

As corollaries, we bound the Lévy and CDF distances between $f(\nu_{pn})$, $f(\mu_p)$ and $f(\mathcal{G}_p)$:

▶ **Corollary 2.2.** *Let f be a harmonic multilinear polynomial of degree d such that with respect to ν_{pn}, $\mathbb{V}[f] \leq 1$ and $\mathrm{Inf}_i^s[f] \leq \tau$ for all $i \in [n]$. There are parameters X_p, X such that for any $0 < \epsilon < 1/2$, if $\tau \leq X_p^{-d}\epsilon^X$ and $n \geq X_p^d/\epsilon^X$ then the Lévy distance between $f(\nu_{pn})$ and $f(\pi)$ is at most ϵ, for $\pi \in \{\mathcal{G}_p, \mu_p\}$. In other words, for all σ,*

$$\Pr_{\nu_{pn}}[f \leq \sigma - \epsilon] - \epsilon \leq \Pr_{\pi}[f \leq \sigma] \leq \Pr_{\nu_{pn}}[f \leq \sigma + \epsilon] + \epsilon.$$

▶ **Corollary 2.3.** *Let f be a harmonic multilinear polynomial of degree d such that with respect to ν_{pn}, $\mathbb{V}[f] = 1$ and $\mathrm{Inf}_i^s[f] \leq \tau$ for all $i \in [n]$. There are parameters Y_p, Y such that for any $0 < \epsilon < 1/2$, if $\tau \leq (Y_p d)^{-d}\epsilon^{Yd}$ and $n \geq (Y_p d)^d/\epsilon^{Yd}$ then the CDF distance between $f(\nu_{pn})$ and $f(\pi)$ is at most ϵ, for $\pi \in \{\mathcal{G}_p, \mu_p\}$. In other words, for all σ,*

$$|\Pr_{\nu_{pn}}[f \leq \sigma] - \Pr_{\pi}[f \leq \sigma]| \leq \epsilon.$$

The proofs of these corollaries closely follows the proof of the analogous results in [11].

2.3 Applications

As applications to our invariance principle, we prove analogues of three classical results in analysis of Boolean functions: Majority is stablest; Bourgain's theorem; and the Kindler–Safra theorem:

▶ **Theorem 2.4.** *Let* $f\colon \binom{[n]}{pn} \to [0,1]$ *have expectation* μ *and satisfy* $\mathrm{Inf}_i^s[f] \leq \tau$ *for all* $i \in [n]$*. For any* $0 < \rho < 1$*, we have*

$$\mathbb{S}_\rho^s[f] \leq \Gamma_\rho(\mu) + O_{p,\rho}\left(\frac{\log\log\frac{1}{\alpha}}{\log\frac{1}{\alpha}}\right) + O_\rho\left(\frac{1}{n}\right), \text{ where } \alpha = \min(\tau, \tfrac{1}{n}),$$

where $\Gamma_\rho(\mu)$ *is the probability that two* ρ*-correlated Gaussians be at most* $\Phi^{-1}(\mu)$ *(here* Φ *is the CDF of a standard Gaussian).*

▶ **Theorem 2.5.** *Fix* $k \geq 2$*. Let* $f\colon \binom{[n]}{pn} \to \{\pm 1\}$ *satisfy* $\mathrm{Inf}_i^s[f^{\leq k}] \leq \tau$ *for all* $i \in [n]$*. For some constants* $W_{p,k}, C$*, if* $\tau \leq W_{p,k}^{-1}\mathbb{V}[f]^C$ *and* $n \geq W_{p,k}/\mathbb{V}[f]^C$ *then*

$$\|f^{>k}\|^2 = \Omega\left(\frac{\mathbb{V}[f]}{\sqrt{k}}\right).$$

▶ **Theorem 2.6.** *Fix the parameter* $k \geq 2$*. Let* $f\colon \binom{[n]}{pn} \to \{\pm 1\}$ *satisfy* $\|f^{>k}\|^2 = \epsilon$*. There exists a function* $h\colon \binom{[n]}{pn} \to \{\pm 1\}$ *of degree* k *depending on* $O_{k,p}(1)$ *coordinates (that is, invariant under permutations of all other coordinates) such that*

$$\|f - h\|^2 = O_{p,k}\left(\epsilon^{1/C} + \frac{1}{n^{1/C}}\right),$$

for some constant C*.*

The proof of Theorem 2.4 closely follows its proof in [11]. The proofs of the other two theorems closely follows analogous proofs in [9].

2.4 t-Intersecting families

As an application of our Kindler–Safra theorem, we prove a stability result for t-intersecting families.

First, a few definitions:

- A t-intersecting family $\mathcal{F} \subseteq \binom{[n]}{k}$ is one in which $|A \cap B| \geq t$ for any $A, B \in \mathcal{F}$.
- A t-star is a family of the form $\{A \in \binom{[n]}{k} : A \supseteq J\}$, where $|J| = t$.
- A $(t,1)$-Frankl family is a family of the form $\{A \in \binom{[n]}{k} : |A \cap J| \geq t+1\}$, where $|J| = t+2$.

Ahlswede and Khachatrian [1, 2] proved that if $n > (t+1)(k-t+1)$ and $\mathcal{F}$ is an intersecting family, then $|\mathcal{F}| \leq \binom{n-t}{k-t}$, and furthermore equality holds if and only if $\mathcal{F}$ is a t-star. They also proved that when $n = (t+1)(k-t+1)$ the same upper bound holds, but now equality holds for both t-stars and $(t,1)$-Frankl families.

A corresponding stability result was proved by Friedgut [7]:

▶ **Proposition 2.7** (Friedgut)**.** *Let* $t \geq 1$*,* $k \geq t$*,* $\lambda, \zeta > 0$*, and* $\lambda n < k < (\frac{1}{t+1} - \zeta)n$*. Suppose* $\mathcal{F} \subseteq \binom{[n]}{k}$ *is a* t*-intersecting family of measure* $|\mathcal{F}| = \binom{n-t}{k-t} - \epsilon\binom{n}{k}$*. Then there exists a family* $\mathcal{G}$ *which is a* t*-star such that*

$$\frac{|\mathcal{F} \triangle \mathcal{G}|}{\binom{n}{k}} = O_{t,\lambda,\zeta}(\epsilon).$$

Friedgut's theorem requires k/n to be bounded away from $1/(t+1)$. Using the Kindler–Safra theorem on the slice rather than the Kindler–Safra theorem on the Boolean cube (which is what Friedgut uses), we can do away with this limitation:

▶ **Theorem 2.8.** *Let $t \geq 2$, $k \geq t+1$ and $n = (t+1)(k-t+1)+r$, where $r > 0$. Suppose that $k/n \geq \lambda$ for some $\lambda > 0$. Suppose $\mathcal{F} \subseteq \binom{[n]}{k}$ is a t-intersecting family of measure $|\mathcal{F}| = \binom{n-t}{k-t} - \epsilon\binom{n}{k}$. Then there exists a family $\mathcal{G}$ which is a t-star or a $(t,1)$-Frankl family such that*

$$\frac{|\mathcal{F} \triangle \mathcal{G}|}{\binom{n}{k}} = O_{t,\lambda}\left(\max\left(\left(\frac{k}{r}\right)^{1/C}, 1\right)\epsilon^{1/C} + \frac{1}{n^{1/C}}\right),$$

for some constant C.

Furthermore, there is a constant $A_{t,\lambda}$ such that $\epsilon \leq A_{t,\lambda}\min(r/k,1)^{C+1}$ implies that $\mathcal{G}$ is a t-star.

Our proof closely follows the argument of Friedgut [7], transplanting it from the setting of the Boolean cube to the setting of the slice, using calculations of Wilson [12] in the latter setting. The argument involves certain subtelties peculiar to the slice.

2.5 Non-harmonic functions

All results we have described so far apply only to harmonic multilinear polynomials. We mentioned that some of these results trivially don't hold for some non-harmonic multilinear polynomials: for example, $\sum_{i=1}^n x_i - np$ doesn't exhibit invariance. This counterexample, however, is a function depending on all coordinates. In contrast, we can show that some sort of invariance does apply for general multilinear polynomials that depend on a small number of coordinates:

▶ **Theorem 2.9.** *Let f be a multilinear polynomial depending on d variables, and let $\tilde{f}$ be the unique harmonic multilinear polynomial agreeing with f on $\binom{[n]}{pn}$, where $d \leq pn \leq n/2$. For $\pi \in \{\mu_p, \mathcal{G}_p\}$ we have*

$$\|f - \tilde{f}\|_\pi^2 = O\left(\frac{d^2 2^d}{p(1-p)n}\right)\|f\|_\pi^2.$$

Proof sketch. Direct calculation shows that if ω is a Fourier character than

$$\|\omega - \tilde{\omega}\|_{\mu_p}^2 = \|\omega - \tilde{\omega}\|_{\mathcal{G}_p}^2 = O\left(\frac{d^2}{p(1-p)n}\right),$$

where $\tilde{\omega}$ is defined analogously to $\tilde{f}$.

We can assume without loss of generality that f depends only on the variables in $[d] = \{1,\ldots,d\}$. Since $\tilde{f} = \sum_{S\subseteq[d]} \hat{f}(S)\tilde{\omega}_S$,

$$\|f - \tilde{f}\|_\pi^2 \leq 2^d \sum_{S\subseteq[d]} \hat{f}(S)^2 O\left(\frac{d^2}{p(1-p)n}\right) = O\left(\frac{d^2 2^d}{p(1-p)n}\right)\|f\|_\pi^2,$$

using the Cauchy–Schwartz inequality. ◀

The idea of the proof is to prove a similar results for Fourier characters for individual Fourier characters, and then to invoke the Cauchy–Schwartz inequality.

As a consequence, if we have a multilinear polynomial f depending on a small number of variables, its harmonic projection $\tilde{f}$ (defined as in the theorem) has a similar expectation, L2 norm, variance and noise stability. This implies, for example, that our Majority is stablest theorem is tight: the harmonic projection of the majority of a small number of indices serves as the tight example.

3 Open problems

Our work gives rise to several open questions.

1. Prove (or refute) an invariance principle comparing ν_{pn} and $\gamma_{p,p}$ for arbitrary (non-harmonic) multilinear polynomials.
2. Prove a tight version of the Kindler–Safra theorem on the slice (Theorem 2.6).
3. The uniform distribution on the slice is an example of a negatively associated vector of random variables. Generalize the invariance principle to this setting.
4. The slice $\binom{[n]}{k}$ can be thought of as a 2-coloring of $[n]$ with a given histogram. Generalize the invariance principle to c-colorings with given histogram.
5. The slice $\binom{[n]}{k}$ has a q-analog: all k-dimensional subspaces of $\mathbb{F}_q^n$ for some prime power q. The analog of the Boolean cube consists of all subspaces of $\mathbb{F}_q^n$ weighted according to their dimension. Generalize the invariance principle to the q-analog, and determine the analog of Gaussian space.

Acknowledgements. This paper started its life when all authors were members of a semester-long program on "Real Analysis in Computer Science" at the Simons Institute for Theory of Computing at U.C. Berkeley. The authors would like to thank the institute for enabling this work.

Y.F. would like to mention that this material is based upon work supported by the National Science Foundation under agreement No. DMS-1128155. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors, and do not necessarily reflect the views of the National Science Foundation. The bulk of the work on this paper was done while at the Institute for Advanced Study, Princeton, NJ.

E.M. would like to acknowledge the support of the following grants: NSF grants DMS 1106999 and CCF 1320105, DOD ONR grant N00014-14-1-0823, and grant 328025 from the Simons Foundation.

K.W. would like to acknowledge the support of NSF grant CCF 1117079.

References

1 Rudolf Ahlswede and Levon H. Khachatrian. The complete intersection theorem for systems of finite sets. *Eur. J. Comb.*, 18(2):125–136, 1997.

2 Rudolf Ahlswede and Levon H. Khachatrian. A pushing-pulling method: New proofs of intersection theorems. *Combinatorica*, 19(1):1–15, 1999.

3 Roee David, Irit Dinur, Elazar Goldenberg, Guy Kindler, and Igor Shinkar. Direct sum testing. In *ITCS 2015*, 2015.

4 Charles F. Dunkl. A Krawtchouk polynomial addition theorem and wreath products of symmetric groups. *Indiana Univ. Math. J.*, 25:335–358, 1976.

5 Charles F. Dunkl. Orthogonal functions on some permutation groups. In *Relations between combinatorics and other parts of mathematics*, volume 34 of *Proc. Symp. Pure Math.*, pages 129–147, Providence, RI, 1979. Amer. Math. Soc.

6 Yuval Filmus. An orthogonal basis for functions over a slice of the boolean hypercube. *Elec. J. Comb.*, 23(1):P1.23, 2016.

7 Ehud Friedgut. On the measure of intersecting families, uniqueness and stability. *Combinatorica*, 28(5):503–528, 2008. doi:10.1007/s00493-008-2318-9.

8 Subhash Khot. Inapproximability of NP-complete problems, discrete fourier analysis, and geometry. In *Proceedings of the International Congress of Mathematicians*, Hyderabad, India, 2010.

9 Guy Kindler, Naomi Kirshner, and Ryan O'Donnell. Gaussian noise sensitivity and Fourier tails, 2014. Manuscript.

10 Elchanan Mossel and Yuval Filmus. Harmonicity and invariance on slices of the Boolean cube. In *31st Conf. Comp. Comp.*, 2016.

11 Elchanan Mossel, Ryan O'Donnell, and Krzysztof Oleszkiewicz. Noise stability of functions with low influences: Invariance and optimality. *Ann. Math.*, 171:295–341, 2010.

12 Richard M. Wilson. The exact bound in the Erdős-Ko-Rado theorem. *Combinatorica*, 4:247–257, 1984.

Harmonicity and Invariance on Slices of the Boolean Cube

Yuval Filmus[1] **and Elchanan Mossel**[2]

1 **Department of Computer Science, Technion – Israel Institute of Technology, Haifa, Isreal**
yuvalfi@cs.technion.ac.il

2 **The Wharton School, University of Pennsylvania, Philadelphia, USA; and University of California, Berkeley, USA**
mossel@wharton.upenn.edu

Abstract

In a recent work with Kindler and Wimmer we proved an invariance principle for the slice for low-influence, low-degree functions. Here we provide an alternative proof for *general* low-degree functions, with no constraints on the influences. We show that any real-valued function on the slice, whose degree when written as a harmonic multi-linear polynomial is $o(\sqrt{n})$, has approximately the same distribution under the slice and cube measure.

Our proof is based on a novel decomposition of random increasing paths in the cube in terms of martingales and reverse martingales. While such decompositions have been used in the past for stationary reversible Markov chains, ours decomposition is applied in a non-reversible non-stationary setup. We also provide simple proofs for some known and some new properties of harmonic functions which are crucial for the proof.

Finally, we provide independent simple proofs for the known facts that 1) one cannot distinguish between the slice and the cube based on functions of $o(n)$ coordinates and 2) Boolean symmetric functions on the cube cannot be approximated under the uniform measure by functions whose sum of influences is $o(\sqrt{n})$.

1998 ACM Subject Classification F.0 [Theory of Computation] General

Keywords and phrases analysis of boolean functions, invariance principle, Johnson association scheme, the slice

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.16

1 Introduction

The basic motivating question for our work is the following:

▶ **Question 1.1.** *Assume n is even. How distinguishable are the uniform measure μ on $\{0,1\}^n$ and the measure ν given by the uniform measure on $\{0,1\}^n$ conditioned on $\sum_i x_i = n/2$?*

More generally: how distinguishable are the product measure μ_p on $\{0,1\}^n$ where each coordinate takes the value 1 independently with probability p and ν_{pn} given by the uniform measure on $\{0,1\}^n$ conditioned on $\sum_i x_i = pn$ (assuming pn is an integer)?

Note that the two measures are easily distinguished using the simple sum of coordinates test. However, our interest is in understanding if the two measures are distinguishable using restricted families of tests, such as *low-depth circuits* or *low-degree polynomials*.

We call $\{0,1\}^n$ the *cube*, the support of the distribution ν_{pn} the *slice*, and the support of ν the *middle slice*. For exposition purposes, the introduction will only discuss the middle

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 16; pp. 16:1–16:13
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

slice, though all results (previous and ours) extend for the case of μ_p and ν_{pn} for every fixed p.

1.1 Low-degree polynomials

In a recent joint work with Kindler and Wimmer [6] we provided a partial answer to Question 1.1 by extending the non-linear invariance principle of [14]. Suppose that f is a low-degree low-influence multilinear polynomial which satisfies $\sum_{i=1}^{n} \frac{\partial f}{\partial x_i} = 0$ (such polynomials are called *harmonic*[1]). The invariance principle of [6] establishes that the distribution of f under the measure ν is close to its distribution under the product measure μ on $\{0,1\}^n$, as well as to its distribution under the product space $\mathbb{R}^n$ equipped with the product Gaussian measure $\mathcal{G} = \mathrm{N}(1/2, 1/4)^{\otimes n}$.

The restriction to multilinear harmonic functions is quite natural in the slice — as every function on the slice has a unique representation as a harmonic multilinear function (this fact, due to Dunkl [3], is proved in Section 3). It is the analog of the implicit restriction to multilinear polynomial in the original non-linear invariance principle.

Both the invariance principle proven in [14] and the one proven in [6] require that the functions have low influences. Indeed, a function like x_1 has a rather different distribution under μ compared to $\mathcal{G}$. Similarly the function $x_1 - x_2$ has a rather different distribution under ν compared to $\mathcal{G}$.

However, note that the distribution of $x_1 - x_2$ under ν is quite similar to its distribution under μ. It is natural to speculate that low-degree harmonic functions have similar distributions under ν and μ. Unfortunately, the proof of the invariance principle in [6] goes through Gaussian space, rendering the low-influence condition necessary even when comparing ν and μ.

Our main result in this paper is a direct proof of the invariance principle on the slice showing that the distribution of a low-degree harmonic function on the slice is close to its distribution on the corresponding cube. Our results do not require the condition of low influences.

▶ **Theorem 1.2.** *Let $f\colon \{-1,1\}^n \to \mathbb{R}$ be a harmonic multilinear polynomial of degree $o(\sqrt{n})$ and variance 1. Then for any 1-Lipschitz function φ (i.e., one satisfying $|\varphi(x) - \varphi(y)| \leq |x - y|$),*

$$|\mathop{\mathbb{E}}_{\nu}[\varphi(f)] - \mathop{\mathbb{E}}_{\mu}[\varphi(f)]| = o(1),$$

and the Lévy distance[2] between the distribution of f under μ and the distribution of f under ν is $o(1)$.

See Section 2 for the definition of harmonic functions and Theorem 4.1 as well as Corollary 4.2 for a more quantitative bounds and more general statements (which apply in particular for any i.i.d. measure on the cube and the corresponding slice). In Subsection 4.1 we show that the results cannot be extended to polynomials whose degree is much bigger than $\sqrt{n}$.

---

[1] This somewhat unfortunate terminology is borrowed from Bergeron [1, Section 8.4], in which an S_n-harmonic polynomial is one which is annihilated by $\sum_{i=1}^{n} \frac{\partial^k}{\partial x_n^k}$ for all k. For multilinear polynomials, both definitions coincide.

[2] The Lévy distance between two real random variables X, Y is the infimal τ such that for all $x \in \mathbb{R}$ it holds that $\Pr[X \leq x - \tau] - \tau \leq \Pr[Y \leq x] \leq \Pr[X \leq x + \tau] + \tau$.

Novel proof ingredients

The basic idea of the proof is to use the coupling method by showing that the distribution of f on different slices of the cube is at most indentical, as long as the slices are of distance at most roughly $\sqrt{n}$ (here the distance between the kth slice and the ℓth slice is $|k-\ell|$). In fact, for our proof to work we crucially need to allow distances which are somewhat larger than $\sqrt{n}$.

To construct the coupling, we use a uniform random increasing path to couple level ℓ to level k above it. The main novel technique is representing the difference between the two levels as a difference of two martingales. Such representations have been used before in the analysis of *stationary reversible* Markov chains in Banach spaces [15], and even earlier in the analysis of stochastic integrals [13]. However, all previous decompositions were for stationary reversible chains while ours is neither. Our novel representation of the differences might be of interest in other applications.

Properties of harmonic functions

Complementing the analytic techniques, we also provide an elegant self-contained treatment of harmonic functions on the slice which is independent, simpler, and contains more results than the first author's earlier paper [5]. In particular we make crucial use of a two-sided Poincaré inequality, see e.g. Lemma 3.11.

Applications

Except for the natural interpretation of Theorem 1.2 in terms of distinguishing between distributions, it can be used to prove results in extremal combinatorics in the same way the main result of Theorem [6] is used. For example, in Proposition 4.5 we give a proof of the Kindler–Safra theorem on the slice, first proved in [6].

1.2 Influences, symmetric functions and circuits

We prove a few other results that give partial answers to Question 1.1.

- Using direct computation of the total variation distance we prove the following theorem:
  ▶ **Theorem 1.3.** *Let f be a function on $\{0,1\}^n$ depending on $o(n)$ coordinates and satisfying $\|f\|_\infty \le 1$. Then*
  $$|\mathop{\mathbb{E}}_{\nu}[f] - \mathop{\mathbb{E}}_{\mu}[f]| = o(1).$$
- We prove that symmetric functions cannot be approximated by functions whose total influence is $o(\sqrt{n})$.
  ▶ **Theorem 1.4.** *There exists a constant $\delta > 0$ such that if f is a symmetric Boolean function such that $\frac{1}{3} \le \mathbb{E}_{\mu_p}[f] \le \frac{2}{3}$ then $\Pr_{\mu_p}[f \neq g] > \delta$ for every Boolean function g satisfying $\mathrm{Inf}[g] = o(\sqrt{n})$.*
  Since it is well-known [2] based on arguments from [12, 7] that a depth d size m circuit has total influence at most $O((\log m)^{d-1})$, our result immediately implies circuit lower bounds for such function. However, much better bounds are known, see e.g. [18] for general symmetric functions and [16] for the case of the majority function. Nevertheless, Theorem 1.4 is more general as it holds for functions f that are not necessarily the majority function and for functions g that are not necessarily in AC^0. Moreover, the proof of Theorem 1.4 is based on a new and simple probabilistic argument.

  Exact formulations and proofs of these results appear in the full version of the paper.

1.3 Other results comparing the cube to the slice

Question 1.1 is not a new question. So we conclude the introduction with a few classical results related to this question:

- The limiting behavior of the partial sum $W(s) = \frac{1}{\sqrt{s}} \sum_{i=1}^{s} (x_i - \frac{1}{2})$ as $s \to \infty$ under the cube and the slice measures are well-studied. It is well-known that under the cube measure $W(s)$ converges to brownian motion, while under the slice measure it converges to a brownian bridge.
- It is well-known that the partial sums $W(s)$ are at least as concentrated in the slice as they are in the cube [8].
- It is well-known that Lipchitz function of the random variables $x_1, \ldots, x_n$ are concentrated both in the cube and in the slice. The results for the slice follow from the hypercontractive estimates by Lee and Yau [11]. These are also needed in our proofs.

Paper organization

A few useful definitions appear in Section 2. Harmonic functions are defined and analyzed in Section 3. We outline the proof of our invariance principle for Lipschitz functions in Section 4, in which we also give an invariance principle for the function $\varphi(x) = (|x| - 1)^2$ and illustrate it by a proof of a Kindler–Safra theorem for the slice, first proved in [6].

Many proofs have been ommitted from this extended abstract. Complete proofs appear in the full version of the paper, available online at `http://arxiv.org/abs/1507.02713`.

2 Definitions

Notation

We employ the falling power notation $n^{\underline{k}} = n(n-1)\cdots(n-k+1)$. The notation $\mathbf{1}_E$ equals 1 if the condition E holds, and 0 otherwise. The sign function is denoted sgn. The *L2 triangle inequality* is $(a+b)^2 \le 2(a+b)^2$ or its generalization $(\sum_{i=1}^{n} a_i)^2 \le n \sum_{i=1}^{n} a_i^2$.

A monomial is *squarefree* if it is not divisible by a square of a variable. (Thus there are 2^n squarefree monomials on n variables.) A polynomial is *multilinear* if all monomials are squarefree. A polynomial is *homogeneous* if all monomials have the same total degree. The dth homogeneous part of a polynomial $f = \sum c_m m$, denote $f^{=d}$, is the sum of $c_m m$ over all monomial m of total degree d. A polynomial f over $x_1, \ldots, x_n$ is *harmonic* if $\sum_{i=1}^{n} \frac{\partial f}{\partial x_i} = 0$.

A univariate function f is *C-Lipschitz* if $|f(x) - f(y)| \le C|x - y|$. A function is *Lipschitz* if it is 1-Lipschitz.

The expectation and variance of a random variable are denoted $\mathbb{E}, \mathbb{V}$, and $\|\cdot\|$ denotes its L2 norm $\|X\| = \sqrt{\mathbb{E}[X^2]}$. To signify that expectation is taken with respect to a distribution α, we write $\mathbb{E}_\alpha[X]$, $\mathbb{V}_\alpha[x]$, and $\|\cdot\|_\alpha$. A normal distribution with mean μ and variance σ^2 is denoted $\mathrm{N}(\mu, \sigma^2)$. A binomial distribution with n trials and success probability p is denoted $\mathrm{B}(n, p)$.

The symmetric group on $[n] = \{1, \ldots, n\}$ is denoted S_n. A distribution on $\mathbb{R}^n$ is *exchangeable* if it is invariant under the action of S_n (that is, under permutation of the coordinates); a discrete distribution is exchangeable if the probability of $(x_1, \ldots, x_n)$ depends only on $x_1 + \cdots + x_n$. For a function f on $\mathbb{R}^n$ and a permutation π, we define $f^\pi(x_1, \ldots, x_n) = f(x_{\pi(1)}, \ldots, x_{\pi(n)})$.

The slice

The n-dimensional Boolean cube is the set $\{0,1\}^n$. For an integer $0 \leq k \leq n$, the kth slice of the n-dimensional Boolean cube is the set

$$\binom{[n]}{k} = \left\{ (x_1, \ldots, x_n) \in \{0,1\}^n : \sum_{i=1}^n x_i = k \right\}.$$

Probability measures

Our work involves two main probability measures, where n is always understood:

- The uniform measure on the slice $\binom{[n]}{k}$ is ν_k.
- The product measure μ_p on the Boolean cube is given by $\mu_p(x) = p^{\sum_i x_i}(1-p)^{\sum_i (1-x_i)}$.

Note that $\nu_k, \mu_{k/n}$ have the same marginal distributions.

3 Harmonic functions

A basic and easy result states that every function on $\{-1,1\}^n$ has a unique representation as a multilinear polynomial, known as the *Fourier expansion*. It is easy to see that a multilinear polynomial has the same mean and variance with respect to the uniform measure on $\{-1,1\}^n$ and with respect to the standard n-dimensional Gaussian measure. In this section we describe the corresponding canonical representation on the slice, due to Dunkl [3, 4] and elaborated by Srinivasan [17] and Filmus [5].

Every function on the slice $\binom{[n]}{k}$ can be represented as a multilinear polynomial, but this representation is not unique. However, as found by Dunkl [3, 4], we can make it unique by demanding that it be *harmonic* in the sense of the following definition.

▶ **Definition 3.1.** A polynomial P over $x_1, \ldots, x_n$ is *harmonic* if

$$\sum_{i=1}^n \frac{\partial P}{\partial x_i} = 0.$$

In other words, P is harmonic if $\Delta P = 0$, where Δ is the differential operator $\sum_{i=1}^n \frac{\partial}{\partial x_i}$.

▶ **Definition 3.2.** A *basic function* is a (possibly empty) product of factors $x_i - x_j$ on disjoint indices. A function is *elementary* if it is a linear combination of basic functions.

Most, but not all, of the harmonic polynomials we consider will be multilinear. Here are some basic properties of harmonic polynomials.

▶ **Lemma 3.3.** *The set of harmonic polynomials is an algebra of polynomials, and is closed under partial derivatives, under permutations of the coordinates, and under taking homogeneous parts. In particular, all elementary functions are harmonic.*

Proof. Suppose f, g are harmonic. Then $\Delta(\alpha f + \beta g) = \alpha\Delta f + \beta\Delta g = 0$; $\Delta(fg) = f\Delta g + g\Delta f = 0$; $\Delta \frac{\partial f}{\partial x_i} = \frac{\partial \Delta f}{\partial x_i} = 0$; and $\Delta(f^\pi) = (\Delta f)^\pi = 0$. Finally, since $\Delta(\sum_{d=0}^n f^{=d}) = \sum_{d=0}^n \Delta f^{=d}$ and $\Delta f^{=d}$ is homogeneous of degree $d-1$, we see that $\Delta f^{=d} = 0^{=d-1} = 0$. ◀

▶ **Lemma 3.4.** *A polynomial f is harmonic if and only if for all $x_1, \ldots, x_n, c$ we have*

$$f(x_1 + c, \ldots, x_n + c) = f(x_1, \ldots, x_n).$$

Proof. Given $x_1, \ldots, x_n$, define a function

$$\phi(x_1, \ldots, x_n, c) = f(x_1 + c, \ldots, x_n + c).$$

The chain rule implies that $\frac{\partial \phi}{\partial c} = \Delta f$. Hence $\Delta f = 0$ iff ϕ is independent of c. ◀

Our first theorem states that every function on the slice has a unique representation as a harmonic multilinear polynomial of degree at most $\min(k, n-k)$.

▶ **Theorem 3.5.** *Let $0 \leq k \leq n$. Every function on the slice $\binom{[n]}{k}$ has a unique representation as a harmonic multilinear polynomial of degree at most* $\min(k, n-k)$.

Similarly, we can prove that every harmonic multilinear polynomial is elementary.

▶ **Lemma 3.6.** *A multilinear polynomial is harmonic iff it is elementary. In particular, a harmonic multilinear polynomial over $x_1, \ldots, x_n$ has degree at most $n/2$.*

As we stated in the introduction to this section, multilinear polynomials enjoy the useful property of having the same mean and variance with respect to all product measures with fixed marginal mean and variance. The corresponding property for harmonic multilinear polynomials is stated in the following theorem, which also follows from the work of the first author [5].

▶ **Theorem 3.7.** *Let f, g be homogeneous harmonic multilinear polynomials of degree d_f, d_g, respectively, and let α be an exchangeable measure. If $d_f \neq d_g$ then $\mathbb{E}_\alpha[fg] = 0$. If $d_f = d_g = d$ then there exists a constant $C_{f,g}$ independent of α such that*

$$\mathop{\mathbb{E}}_{\alpha}[fg] = C_{f,g} \mathop{\mathbb{E}}_{\alpha}[(x_1 - x_2)^2 \cdots (x_{2d-1} - x_{2d})^2].$$

▶ **Corollary 3.8.** *Let f be a harmonic multilinear polynomial of degree at most d with constant coefficient $f^{=0}$. Suppose that α, β are exchangeable measures and $C > 0$ is a constant that for $t \leq d$ satisfies*

$$\mathop{\mathbb{E}}_{\alpha}[(x_1 - x_2)^2 \cdots (x_{2t-1} - x_{2t})^2] \leq C \mathop{\mathbb{E}}_{\beta}[(x_1 - x_2)^2 \cdots (x_{2t-1} - x_{2t})^2].$$

Then $\mathbb{E}_\alpha[f] = f^{=0}$, $\|f\|_\alpha^2 \leq C\|f\|_\beta^2$, and $\mathbb{V}_\alpha[f] \leq C\,\mathbb{V}_\beta[f]$.

The following lemma computes $\mathbb{E}[(x_1 - x_2)^2 \cdots (x_{2d-1} - x_{2d})^2]$ for the measures ν_k, μ_p.

▶ **Lemma 3.9.** *Let $p = k/n$. We have*

$$\mathop{\mathbb{E}}_{\nu_k}[(x_1 - x_2)^2 \cdots (x_{2d-1} - x_{2d})^2] = 2^d \frac{k^{\underline{d}}(n-k)^{\underline{d}}}{n^{\underline{2d}}} = (2p(1-p))^d \left(1 \pm O\left(\frac{d^2}{p(1-p)n}\right)\right),$$
$$\mathop{\mathbb{E}}_{\mu_p}[(x_1 - x_2)^2 \cdots (x_{2d-1} - x_{2d})^2] = (2p(1-p))^d.$$

This straightforward computation appears in [5, Theorem 4.1] and [6, Lemma 2.9]. Qualitatively, the lemma states that the norm of a low degree basic function is similar in both ν_k and μ_p. This is not surprising: the coordinates in the slice are almost independent, and a low degree basic function depends only on a small number of them.

We proceed by stating the so-called two-sided Poincaré inequality, starting with the following fact.

▶ **Lemma 3.10.** *Let f be a harmonic multilinear polynomial. Then*

$$\sum_{i<j} f^{(i\,j)} = \sum_{d=0}^{n/2} \left[\binom{n}{2} - d(n-d+1)\right] f^{=d},$$

where $f^{=d}$ is the dth homogeneous part of f.

▶ **Lemma 3.11.** *Let f be a harmonic multilinear polynomial of degree at most d. Then with respect to any exchangeable measure,*

$$n\,\mathbb{V}[f] \leq \frac{1}{2}\sum_{i<j} \|f - f^{(i\,j)}\|^2 \leq d(n-d+1)\,\mathbb{V}[f].$$

Finally, we state another two-sided Poincaré inequality, this time for derivatives. We start with the following surprising corollary of Theorem 3.7.

▶ **Lemma 3.12.** *Let f, g be homogeneous harmonic multilinear polynomials of degree d. Then for any exchangeable measure α,*

$$\frac{\sum_{i=1}^n \mathbb{E}_\alpha\left[\frac{\partial f}{\partial x_i}\frac{\partial g}{\partial x_i}\right]}{\mathbb{E}_\alpha[fg]} = 2d\frac{\mathbb{E}_\alpha[(x_1-x_2)^2\cdots(x_{2d-3}-x_{2d-2})^2]}{\mathbb{E}_\alpha[(x_1-x_2)^2\cdots(x_{2d-1}-x_{2d})^2]}.$$

We deduce the following two-sided Poincaré inequality.

▶ **Lemma 3.13.** *Let f be a multilinear polynomial of degree d, and let α be an exchangeable measure. Suppose that for $1 \leq t \leq d$ we have*

$$m \leq 2t\frac{\mathbb{E}_\alpha[(x_1-x_2)^2\cdots(x_{2t-3}-x_{2t-2})^2]}{\mathbb{E}_\alpha[(x_1-x_2)^2\cdots(x_{2t-1}-x_{2t})^2]} \leq M.$$

Then also

$$m\,\mathbb{V}[f] \leq \sum_{i=1}^n \left\|\frac{\partial f}{\partial x_i}\right\|^2 \leq M\,\mathbb{V}[f].$$

The following lemma computes m, M for the measures ν_k, μ_p.

▶ **Lemma 3.14.** *Let $p = k/n$. We have*

$$\begin{aligned}
2d\frac{\mathbb{E}_{\nu_k}[(x_1-x_2)^2\cdots(x_{2d-3}-x_{2d-2})^2]}{\mathbb{E}_{\nu_k}[(x_1-x_2)^2\cdots(x_{2d-1}-x_{2d})^2]} &= d\frac{(n-2d+2)(n-2d+1)}{(k-d+1)(n-k-d+1)} \\
&= \frac{d}{p(1-p)}\left(1 \pm O\left(\frac{d}{p(1-p)n}\right)\right), \\
2d\frac{\mathbb{E}_{\mu_p}[(x_1-x_2)^2\cdots(x_{2d-3}-x_{2d-2})^2]}{\mathbb{E}_{\mu_p}[(x_1-x_2)^2\cdots(x_{2d-1}-x_{2d})^2]} &= \frac{d}{p(1-p)}.
\end{aligned}$$

4 Invariance principle

In this section we state and outline the proof of an invariance principle showing that the distribution of a low-degree function on a slice $\binom{[n]}{k}$ is similar to its distribution on the Boolean cube with respect to the measure $\mu_{k/n}$. For convenience, we analyze the similarity in distribution via Lipschitz test functions, and derive similarity in more conventional terms as a corollary. The basic idea is to show that the distribution of a low degree function on a

given slice $\binom{[n]}{k}$ is similar to its distribution on nearby slices $\binom{[n]}{\ell}$. If we can show this for all slices satisfying $|k-\ell| \leq B$ for some $B = \omega(\sqrt{n})$, then the invariance follows by decomposing the Boolean cube as a union of slices.

Here is the formal statement of our invariance principle.

▶ **Theorem 4.1.** *There exists a constant $K > 0$ such that the following holds.*

Let f be a harmonic multilinear polynomial of degree d satisfying $d^2 \leq K\frac{p(1-p)n}{\log[p(1-p)n]}$ such that $\mathbb{V}[f]_{\nu_{pn}} = 1$. For any Lipschitz function φ,

$$|\mathop{\mathbb{E}}_{\nu_{pn}}[\varphi(f)] - \mathop{\mathbb{E}}_{\mu_p}[\varphi(f)]| = O\left(\sqrt{\frac{d}{\sqrt{p(1-p)n}}\log^{1/2}\frac{\sqrt{p(1-p)n}}{d}}\right).$$

As a corollary, we can estimate the Lévy distance between $f(\nu_{pn})$ and $f(\mu_p)$, along the lines of [14, Theorem 3.19(28)].

▶ **Corollary 4.2.** *Let f be a harmonic multilinear polynomial of degree d satisfying $d^2 \leq K\frac{p(1-p)n}{\log[p(1-p)n]}$ such that $\mathbb{V}[f]_{\nu_{pn}} = 1$, where $K > 0$ is the constant from Theorem 4.1. The Lévy distance between $f(\nu_{pn})$ and $f(\mu_p)$ is at most*

$$\epsilon = O\left(\sqrt[4]{\frac{d}{\sqrt{p(1-p)n}}\log^{1/2}\frac{\sqrt{p(1-p)n}}{d}}\right).$$

That is, for all y it holds that

$$\Pr_{\nu_{pn}}[f \leq y-\epsilon] - \epsilon \leq \Pr_{\mu_p}[f \leq y] \leq \Pr_{\nu_{pn}}[f \leq y+\epsilon] + \epsilon.$$

We conjecture that $f(\nu_{pn})$ and $f(\mu_p)$ are also close in CDF distance, but unfortunately the method of proof of [14, Theorem 3.19(30)] fails in this case.

The complete proof of Theorem 4.1 appears in the full version of the paper. In the remainder of this section, we explain the intuition behind the proof.

Our argument concerns the following objects:

- A harmonic multilinear polynomial f of degree d and unit norm. We think of d as "small".
- A Lipschitz functional φ.
- A slice $\binom{[n]}{pn}$. We think of p as constant, though the argument even works for subconstant p.

Our goal is to show that $\mathbb{E}_{\mu_p}[\varphi(f)] \approx \mathbb{E}_{\nu_{pn}}[\varphi(f)]$. The first step is to express μ_p as a mixture of ν_ℓ for various ℓ:

$$\mathop{\mathbb{E}}_{\mu_p}[\varphi(f)] = \sum_{\ell=0}^{n}\binom{n}{\ell}p^\ell(1-p)^{n-\ell}\mathop{\mathbb{E}}_{\nu_\ell}[\varphi(f)].$$

Applying the triangle inequality, this shows that

$$|\mathop{\mathbb{E}}_{\mu_p}[\varphi(f)] - \mathop{\mathbb{E}}_{\nu_{pn}}[\varphi(f)]| \leq \sum_{\ell=0}^{n}\binom{n}{\ell}p^\ell(1-p)^{n-\ell}|\mathop{\mathbb{E}}_{\nu_\ell}[\varphi(f)] - \mathop{\mathbb{E}}_{\nu_{pn}}[\varphi(f)]|.$$

In general we expect $|\mathbb{E}_{\nu_\ell}[\varphi(f)] - \mathbb{E}_{\nu_{pn}}[\varphi(f)]|$ to grow with $|\ell - pn|$, and our strategy would be to consider separately slices close to pn, say $|pn-\ell| \leq \delta$, and slices far away from pn, say $|pn-\ell| > \delta$. We will bound the contribution of slices close to pn directly. If δ is large enough then we expect the contribution of slices far away from pn to be small, essentially

since μ_p is concentrated on slices close to pn. For this argument to work, we need to choose δ so that $\delta = \omega(\sqrt{n})$.

It remains to bound $|\mathbb{E}_{\nu_\ell}[\varphi(f)] - \mathbb{E}_{\nu_{pn}}[\varphi(f)]|$ for ℓ close to pn. One strategy to obtain such a bound is to bound instead $|\mathbb{E}_{\nu_s}[\varphi(f)] - \mathbb{E}_{\nu_{s+1}}[\varphi(s)]|$ for various s, and use the triangle inequality. To this end, it is natural to consider the following coupling: let $(\mathbf{X}(s), \mathbf{X}(s+1)) \in \binom{[n]}{s} \times \binom{[n]}{s+1}$ be chosen uniformly at random under the constraint $\mathbf{X}(s) \subset \mathbf{X}(s+1)$. We can then bound

$$|\mathop{\mathbb{E}}_{\nu_s}[\varphi(f)] - \mathop{\mathbb{E}}_{\nu_{s+1}}[\varphi(s)]| = |\mathbb{E}[\varphi(f(\mathbf{X}(s))) - \varphi(f(\mathbf{X}(s+1)))]| \leq \mathbb{E}[|\varphi(f(\mathbf{X}(s))) - \varphi(f(\mathbf{X}(s+1)))|] \leq \mathbb{E}[|f(\mathbf{X}(s)) - f(\mathbf{X}(s+1))|].$$

Denoting $\boldsymbol{\pi}(s+1) = \mathbf{X}(s+1) \setminus \mathbf{X}(s)$ and using the mulilinearity of f, this shows that

$$|\mathop{\mathbb{E}}_{\nu_s}[\varphi(f)] - \mathop{\mathbb{E}}_{\nu_{s+1}}[\varphi(s)]| \leq \mathbb{E}\left[\left|\frac{\partial f}{\partial x_{\boldsymbol{\pi}(s+1)}}(\mathbf{X}(s))\right|\right] = \mathbb{E}\left[\frac{1}{n-s}\sum_{i \notin \mathbf{X}(s)}\left|\frac{\partial f}{\partial x_i}(\mathbf{X}(s))\right|\right].$$

While we cannot bound $\sum_i |\frac{\partial f}{\partial x_i}|$ directly, Lemma 3.13 implies that $\sum_i \left(\frac{\partial f}{\partial x_i}\right)^2 = O(d)$. Applying Cauchy–Schwartz, we get that for s close to pn,

$$\begin{aligned}|\mathop{\mathbb{E}}_{\nu_s}[\varphi(f)] - \mathop{\mathbb{E}}_{\nu_{s+1}}[\varphi(s)]| &\leq \frac{1}{\Theta(n)}\mathbb{E}\left[\sum_{i=1}^n \left|\frac{\partial f}{\partial x_i}(\mathbf{X}(s))\right|\right] \\ &\leq \frac{1}{\Theta(n)}\mathbb{E}\left[\sqrt{n}\sqrt{\sum_{i=1}^n \frac{\partial f}{\partial x_i}(\mathbf{X}(s))^2}\right] = O\left(\sqrt{\frac{d}{n}}\right).\end{aligned}$$

Recall now that our original goal was to bound $|\mathbb{E}_{\nu_\ell}[\varphi(f)] - \mathbb{E}_{\nu_{pn}}[\varphi(f)]|$ for $|\ell - pn| \leq \delta$, and our intended δ satisfied $\delta = \omega(\sqrt{n})$. Unfortunately, the idea just described only gives a bound of the form $|\mathbb{E}_{\nu_\ell}[\varphi(f)] - \mathbb{E}_{\nu_{pn}}[\varphi(f)]| = O(\delta\sqrt{d/n})$, which is useless for our intended δ.

One way out is to take $\delta = C\sqrt{n}$. This allows us to obtain meaningful bounds both on the contribution of slices close to pn and on the contribution of slices far away from pn. Although this only gives a constant upper bound on $|\mathbb{E}_{\mu_p}[\varphi(f)] - \mathbb{E}_{\nu_{pn}}[\varphi(f)]|$ if applied directly, this idea can be used in conjuction with the invariance principle for the Boolean cube [14] to give an invariance principle for the slice, and this is the route chosen in the prequel [6]. One drawback of this approach is that the invariance principle for the Boolean cube requires all influences to be small.

Our approach, in contrast, considers a coupling $(\mathbf{X}(0), \ldots, \mathbf{X}(n))$ of *all* slices. Analogous to $f(\mathbf{X}(s+1)) - f(\mathbf{X}(s))$, we consider the quantity

$$\mathbf{C}(s) = (n-s)(f(\mathbf{X}(s+1)) - f(\mathbf{X}(s))) - s(f(\mathbf{X}(s-1)) - f(\mathbf{X}(s))).$$

As before, we can bound $\mathbb{E}[|\mathbf{C}(s)|] = O(\sqrt{dn})$. Moreover,

$$\sum_{u=s}^t \mathbf{C}(u) = (n-t)f(\mathbf{X}(t+1)) + (t-1)f(\mathbf{X}(t)) - (n-s-1)f(\mathbf{X}(s)) - sf(\mathbf{X}(s-1)),$$

and so we can bound $|\mathbb{E}_{\nu_\ell}[\varphi(f)] - \mathbb{E}_{\nu_{pn}}[\varphi(f)]|$ by bounding the expectation of $\sum_{u=pn}^{\ell} \mathbf{C}(u)$ or of $\sum_{u=\ell}^{pn} \mathbf{C}(u)$. The triangle inequality gives $|\sum_{u=s}^t \mathbf{C}(u)| = O(|s-t|\sqrt{dn})$, which suffers

from the same problem that we encountered above. However, by expressing $\mathbf{C}(s)$ as a difference of two martingales, we are able to improve on the triangle inequality, showing that

$$\left|\sum_{u=s}^{t}\mathbf{C}(u)\right| = O(\sqrt{|s-t|dn}),$$

a bound which is useful for $|s-t| = o(n/d)$ rather than for $|s-t| = o(\sqrt{n/d})$ as before.

In more detail, we define

$$\mathbf{U}(u) = f(\mathbf{X}(u+1)) - f(\mathbf{X}(u)) - \mathbb{E}[f(\mathbf{X}(u+1)) - f(\mathbf{X}(u))|\mathbf{X}(u)],$$
$$\mathbf{D}(u) = f(\mathbf{X}(u-1)) - f(\mathbf{X}(u)) - \mathbb{E}[f(\mathbf{X}(u-1)) - f(\mathbf{X}(u))|\mathbf{X}(u)],$$

both martingales by construction, $\mathbf{U}(u)$ for increasing u, and $\mathbf{D}(u)$ for decreasing u. We claim that $\mathbf{C}(u) = (n-u)\mathbf{U}(u) - u\mathbf{D}(u)$. If this holds, then using the fact that $\mathbb{E}[\mathbf{U}(u)\mathbf{U}(v)] = \mathbb{E}[\mathbf{D}(u)\mathbf{D}(v)] = 0$ for $u \neq v$ and the L2 triangle inequality $(a+b)^2 \leq 2a^2 + 2b^2$, we get

$$\begin{aligned}\mathbb{E}\left[\left(\sum_{u=s}^{t}\mathbf{C}(u)\right)^2\right] &\leq 2\,\mathbb{E}\left[\left(\sum_{u=s}^{t}(n-u)\mathbf{U}(u)\right)^2\right] + 2\,\mathbb{E}\left[\left(\sum_{u=s}^{t}u\mathbf{D}(u)\right)^2\right] \\ &= 2\sum_{u=s}^{t}(n-u)^2\,\mathbb{E}[\mathbf{U}(u)^2] + 2\sum_{u=s}^{t}u^2\,\mathbb{E}[\mathbf{D}(u)^2].\end{aligned}$$

This shows that $\mathbb{E}[(\sum_{u=s}^{t}\mathbf{C}(u))^2]$ scales linearly in $t-s$ rather than quadratically in $t-s$, which is what we would get if we just applied the triangle inequality. Since the L1 norm is bounded by the L2 norm, we conclude that $\mathbb{E}[|\sum_{u=s}^{t}\mathbf{C}(u)|] = O(\sqrt{|s-t|dn})$.

Finally, let us explain why $\mathbf{C}(u) = (n-u)\mathbf{U}(u) - u\mathbf{D}(u)$. In view of our previous expression for $\mathbf{C}(u)$, this boils down to proving that

$$(n-u)\,\mathbb{E}[f(\mathbf{X}(u+1)) - f(\mathbf{X}(u))|\mathbf{X}(u)] - u\,\mathbb{E}[f(\mathbf{X}(u-1)) - f(\mathbf{X}(u))|\mathbf{X}(u)] = 0.$$

We can rewrite the left-hand side as

$$\mathbb{E}\left[\sum_{i\notin\mathbf{X}(u)}[f(\mathbf{X}(u)\cup\{i\}) - f(\mathbf{X}(u))] - \sum_{i\in\mathbf{X}(u)}[f(\mathbf{X}(u)\setminus\{i\}) - f(\mathbf{X}(u))]\right].$$

Since f is multilinear, we can replace the differences with derivatives:

$$\mathbb{E}\left[\sum_{i\notin\mathbf{X}(u)}\frac{\partial f}{\partial x_i}(\mathbf{X}(u)) - \sum_{i\in\mathbf{X}(u)} -\frac{\partial f}{\partial x_i}(\mathbf{X}(u))\right] = \mathbb{E}\left[\sum_{i=1}^{n}\frac{\partial f}{\partial x_i}(\mathbf{X}(u))\right].$$

However, the last expression clearly vanishes, since f is harmonic. This completes the outline of the proof.

4.1 High-degree functions

Theorem 4.1 requires that $d = o(\sqrt{p(1-p)n})$. Indeed, Lemma 3.9, which implies that the norm of a low degree function is approximately the same under both μ_p and ν_{pn}, already requires the degree to be $o(\sqrt{p(1-p)n})$. For $d = \omega(\sqrt{p(1-p)n})$ and constant $p \neq 1/2$ we exhibit below a $0/\pm 1$-valued function f which satisfies $\|f\|_{\mu_p} = 1$ while $\|f\|_{\nu_{pn}} = o(1)$. This shows that for constant $p \neq 1/2$ the dependence on the degree is essential in Theorem 4.1,

since $|\mathbb{E}_{\nu_{pn}}[|f|] - \mathbb{E}_{\mu_p}[|f|]| = \|f\|_{\mu_p}^2 - \|f\|_{\nu_{pn}}^2 = 1 - o(1)$. We do not know whether this dependence is necessary for $p = 1/2$. Indeed, Lemma 3.9 can be extended above $\sqrt{n}$ in this case, as the calculation below shows.

Let $d = \omega(\sqrt{p(1-p)n})$, and assume further that $d = o((p(1-p)n)^{2/3})$. We consider the function $f = (2p(1-p))^{-d/2}(x_1 - x_2)\cdots(x_{2d-1} - x_{2d})$, whose μ_p-norm is 1 according to Lemma 3.9. The lemma also gives its ν_k-norm (where $k = pn$) as

$$\|f\|_{\nu_k}^2 = (p(1-p))^{-d}\frac{k^{\underline{d}}(n-k)^{\underline{d}}}{n^{\underline{2d}}}.$$

We estimate this expression using Stirling's approximation, starting with $k^{\underline{d}}$:

$$k^{\underline{d}} = \frac{k!}{(k-d)!} = \left(\frac{k}{k-d}\right)^{k-d+1/2}\frac{k^d}{e^d}e^{O(1/k)-O(1/(k-d))} = \left(1 + \frac{d}{k-d}\right)^{k-d}\frac{k^d}{e^d}(1 \pm o(1)).$$

The Taylor series $\log(1+x) = x - x^2/2 + O(x^3)$ shows that

$$\left(1 + \frac{d}{k-d}\right)^{k-d} = \exp\left[d - \frac{d^2}{2(k-d)} + o(1)\right] = \exp\left[d - \frac{d^2}{2k} + o(1)\right],$$

and so $k^{\underline{d}} = k^d e^{-d^2/2k}(1 \pm o(1))$. We can similarly estimate $(n-k)^{\underline{d}} = (n-k)^d e^{-d^2/2(n-k)}(1 \pm o(1))$ and $n^{\underline{2d}} = n^{2d}e^{-2d^2/n}(1 \pm o(1))$, concluding that

$$\begin{aligned}\|f\|_{\nu_k}^2 &= (p(1-p))^{-d}\frac{k^d(n-k)^d}{n^{2d}}e^{-d^2/2k - d^2/2(n-k) + 2d^2/n}(1 \pm o(1)) \\ &= \exp\left[\frac{d^2}{2p(1-p)n}(-1 + 4p(1-p)) \pm o(1)\right].\end{aligned}$$

If $p \neq 1/2$ is fixed, we immediately conclude that $\|f\|_{\nu_k} = o(1)$.

4.2 Approximately Boolean functions

Theorem 4.1 only applies to Lipschitz test functions, but in many applications we are interested in functions which grow faster, for example the distance-from-$\{-1,1\}$ function $\varphi(x) = (|x| - 1)^2$. Using hypercontractivity, we can obtain an invariance principle for $\varphi(x) = (|x| - 1)^2$.

▶ **Theorem 4.3.** *Let f be a harmonic multilinear polynomial of degree d satisfying $d^2 \leq K\frac{p(1-p)n}{\log[p(1-p)n]}$ such that $\|f\|_{\nu_{pn}} = 1$, where K is the constant in Theorem 4.1. We have*

$$|\mathop{\mathbb{E}}_{\nu_{pn}}[(|f|-1)^2] - \mathop{\mathbb{E}}_{\mu_p}[(|f|-1)^2]| = O\left(\frac{d^{1/4}\log^{1/8} n}{n^{1/8}}(p(1-p))^{-O(d)}\right).$$

As an illustration of this theorem, we give an alternative proof of [6, Theorem 7.5], a Kindler–Safra theorem for the slice.

▶ **Definition 4.4.** A function f on a given domain is *Boolean* if on the domain it satisfies $f \in \{\pm 1\}$. If the domain is a cube, we use the term *cube-Boolean*. If it is a slice, we use the term *slice-Boolean*.

▶ **Proposition 4.5.** *Let f be a multilinear polynomial of degree d such that $\mathbb{E}_{\mu_p}[(|f|-1)^2] = \epsilon$. There exists a cube-Boolean function g on $(p(1-p))^{-O(d)}$ coordinates such that $\|f - g\|_{\mu_p}^2 = O((p(1-p))^{-O(d)}\epsilon)$.*

Proof. This is essentially proved in [10, 9]. Explicitly, they prove the same result without the guarantee that g is cube-Boolean. In order to get our version, let $F = \operatorname{sgn} f$ and $G = \operatorname{sgn} g$. By definition $\|F - f\|^2 = \epsilon$, and so $\|F - g\|^2 = O((p(1-p))^{-O(d)}\epsilon)$. Since F is cube-Boolean, this implies that $\|F - G\|^2 = O((p(1-p))^{-O(d)}\epsilon)$. We conclude that $\|f - G\|^2 = O((p(1-p))^{-O(d)}\epsilon)$. ◀

▶ **Theorem 4.6.** *For every $d > 0$ there is a parameter $N_{d,p}$ depending continuously on p such that if $n > N_{d,p}$ then the following holds.*

Let f be a slice-Boolean harmonic multilinear polynomial such that $\|f^{>d}\|^2_{\nu_{pn}} = \epsilon$, where $f^{>d} = \sum_{i>d} f^{=i}$. There exists a slice-Boolean harmonic multilinear polynomial g that depends on $(p(1-p))^{-O(d)}$ coordinates (invariant to permutations of the other coordinates) satisfying

$$\|f - g\|^2_{\nu_{pn}} \leq O((p(1-p))^{-O(d)}\epsilon) + \tilde{O}\left(\frac{1}{n^{1/8}}\right),$$

where $\tilde{O}$ hides polylogarithmic factors.

Proof. Let $\tilde{f} = f^{\leq d}$ (that is, $\tilde{f} = \sum_{i \leq d} f^{=i}$), so that $\mathbb{E}_{\nu_{pn}}[(|\tilde{f}| - 1)^2] \leq \mathbb{E}_{\nu_{pn}}[(\tilde{f} - f)^2] = \epsilon$. Notice that $\tilde{f}$ is a harmonic multilinear polynomial of degree at most d. Theorem 4.3 implies that

$$\mathop{\mathbb{E}}_{\mu_p}[(|\tilde{f}| - 1)^2] \leq \underbrace{\epsilon + O\left(\frac{d^{1/4}\log^{1/8} n}{n^{1/8}}(p(1-p))^{-O(d)}\right)}_{\epsilon_1}.$$

Proposition 4.5 implies that there exists a cube-Boolean function g on a set J of $M = O((p(1-p))^{-O(d)})$ coordinates such that $\epsilon_2 \triangleq \mathbb{E}_{\mu_p}[(\tilde{f} - g)^2] = O((p(1-p))^{-O(d)}\epsilon_1)$. The function g is also slice-Boolean, but it is not necessarily harmonic. Let h be the unique harmonic multilinear function of degree at most $\min(k, n-k)$ that agrees with g on $\binom{[n]}{pn}$. Note that h also depends only on the coordinates in J, and in particular it has degree at most M (in fact, [6, Lemma 3.1] implies that $\deg h \leq \deg g$). Invoking [6, Theorem 3.3], we see that $\|g - h\|^2_{\mu_p} = O(\frac{M^2 2^M}{p(1-p)n})$, and so

$$\epsilon_3 = \|\tilde{f} - h\|^2_{\mu_p} = O\left(\frac{M^2 2^M}{p(1-p)n} + \epsilon_2\right).$$

Corollary 3.8 and Lemma 3.9 imply that $\|\tilde{f} - h\|^2_{\nu_{pn}} = O(\epsilon_3)$, using the fact that $\deg(\tilde{f} - h) \leq M$. The proof is completed by noticing that $\|f - h\|^2_{\nu_{pn}} = O(\epsilon + \epsilon_3)$. ◀

The proof of [6, Theorem 7.5] contains an additional argument guaranteeing that $\deg g \leq d$. The same argument can be applied here.

Acknowledgements. Both authors would like to thank the referees for their extensive and helpful comments.

Y.F. would like to mention that this material is based upon work supported by the National Science Foundation under agreement No. DMS-1128155. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors, and do not necessarily reflect the views of the National Science Foundation. Part of the work was done while at the Institute for Advanced Study, Princeton, NJ.

E.M. would like to acknowledge the support of the following grants: NSF grants DMS 1106999 and CCF 1320105, DOD ONR grant N00014-14-1-0823, and grant 328025 from the Simons Foundation.

References

1. Francois Bergeron. *Algebraic Combinatorics and Coinvariant Spaces*. CMS Treatises in Mathematics. A K Peters, 2009.
2. Ravi B Boppana. The average sensitivity of bounded-depth circuits. *Information Processing Letters*, 63(5):257–261, 1997.
3. Charles F. Dunkl. A Krawtchouk polynomial addition theorem and wreath products of symmetric groups. *Indiana Univ. Math. J.*, 25:335–358, 1976.
4. Charles F. Dunkl. Orthogonal functions on some permutation groups. In *Relations between combinatorics and other parts of mathematics*, volume 34 of *Proc. Symp. Pure Math.*, pages 129–147, Providence, RI, 1979. Amer. Math. Soc.
5. Yuval Filmus. An orthogonal basis for functions over a slice of the boolean hypercube. *Elec. J. Comb.*, 23(1):P1.23, 2016.
6. Yuval Filmus, Guy Kindler, Elchanan Mossel, and Karl Wimmer. Invariance principle on the slice. In *31st Conf. Comp. Comp.*, 2016.
7. Johan Håstad. Almost optimal lower bounds for small depth circuits. In Silvio Micali, editor, *Randomness and Computation*, volume 5 of *Advances in Computing Research*, pages 143–170. JAI Press, 1989.
8. Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American statistical association*, 58(301):13–30, 1963.
9. Guy Kindler. *Property testing, PCP and Juntas*. PhD thesis, Tel-Aviv University, 2002.
10. Guy Kindler and Shmuel Safra. Noise-resistant Boolean functions are juntas, 2004. Unpublished manuscript.
11. Tzong-Yau Lee and Horng-Tzer Yau. Logarithmic Sobolev inequality for some models of random walks. *Ann. Prob.*, 26(4):1855–1873, 1998.
12. N. Linial, Y. Mansour, and N. Nisan. Constant depth circuits, fourier transform and learnability. *Journal of the ACM*, 40(3):607–620, 1993.
13. Terry J. Lyons and T. S. Zhang. Decomposition of Dirichlet processes and its application. *Ann. Probab.*, 22(1):494–524, 1994.
14. Elchanan Mossel, Ryan O'Donnell, and Krzysztof Oleszkiewicz. Noise stability of functions with low influences: Invariance and optimality. *Ann. Math.*, 171:295–341, 2010.
15. Assaf Naor, Yuval Peres, Oded Schramm, and Scott Sheffield. Markov chains in smooth Banach spaces and Gromov-hyperbolic metric spaces. *Duke Math J.*, 134(1):165–197, 2006.
16. Ryan O'Donnell and Karl Wimmer. Approximation by DNF: Examples and counterexamples. In *Automata, Languages and Programming*, volume 4596 of *Lecture Notes in Computer Science*, pages 195–206. Springer Berlin Heidelberg, 2007.
17. Murali K. Srinivasan. Symmetric chains, Gelfand–Tsetlin chains, and the Terwilliger algebra of the binary Hamming scheme. *J. Algebr. Comb.*, 34(2):301–322, 2011.
18. Avishay Tal. Tight bounds on the Fourier spectrum of AC^0. Manuscript, 2014.

On the Sum-of-Squares Degree of Symmetric Quadratic Functions

Troy Lee[*1], **Anupam Prakash**[†2], **Ronald de Wolf**[‡3], **and Henry Yuen**[§4]

1 **School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore; and Centre for Quantum Technologies, Singapore**
troyjlee@gmail.com

2 **School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore; and Centre for Quantum Technologies, Singapore**
aprakash@ntu.edu.sg

3 **QuSoft, Amsterdam, The Netherlands; and CWI, Amsterdam, The Netherlands; and University of Amsterdam, Amsterdam, The Netherlands**
rdewolf@cwi.nl

4 **Massachusetts Institute of Technology, Cambridge, USA**
hyuen@mit.edu

Abstract

We study how well functions over the boolean hypercube of the form $f_k(x) = (|x|-k)(|x|-k-1)$ can be approximated by sums of squares of low-degree polynomials, obtaining good bounds for the case of approximation in ℓ_∞-norm as well as in ℓ_1-norm. We describe three complexity-theoretic applications: (1) a proof that the recent breakthrough lower bound of Lee, Raghavendra, and Steurer [19] on the positive semidefinite extension complexity of the correlation and TSP polytopes cannot be improved further by showing better sum-of-squares degree lower bounds on ℓ_1-approximation of f_k; (2) a proof that Grigoriev's lower bound on the degree of Positivstellensatz refutations for the knapsack problem is optimal, answering an open question from [12]; (3) bounds on the query complexity of quantum algorithms whose expected output approximates such functions.

1998 ACM Subject Classification G.1.2 Approximation

Keywords and phrases Sum-of-squares degree, approximation theory, Positivstellensatz refutations of knapsack, quantum query complexity in expectation, extension complexity

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.17

[*] Supported by the Singapore National Research Foundation under NRF RF Award No. NRF-NRFF2013-13.

[†] Supported by the Singapore National Research Foundation under NRF RF Award No. NRF-NRFF2013-13.

[‡] Partially supported by the ERC Consolidator Grant QPROGRESS and by the European Commission FET-Proactive project Quantum Algorithms (QALGO) 600700.

[§] Supported by NSF grants 1218547 and 1452302 and a Simons Graduate Award in Theoretical Computer Science.

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 17; pp. 17:1–17:31

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

1 Introduction

1.1 Approximation of functions on the boolean hypercube by polynomials

Classical approximation theory studies how well a function $f : \mathbb{R} \to \mathbb{R}$ can be approximated by simpler functions, most commonly by polynomials of bounded degree. Approximation theory has found applications throughout complexity theory, for example in learning theory [21, 23], query complexity [22, 1], communication complexity [29, 30], and more.

An important special case is the investigation of the best approximation to a real boolean function $f : \{0,1\}^n \to \mathbb{R}$ in ℓ_∞-distance by a degree-d polynomial in the n variables $x_1, \ldots, x_n$. Nisan and Szegedy [22] initiated this study, showing that any polynomial that approximates the OR function with constant error in ℓ_∞-norm on $\{0,1\}^n$ has degree $\Omega(\sqrt{n})$. They also showed this bound is tight by constructing an $O(\sqrt{n})$-degree approximating polynomial for the OR function from a Chebyshev polynomial. Paturi [25] followed this by characterizing the approximate degree of *any* symmetric boolean function, i.e., any function $f : \{0,1\}^n \to \{0,1\}$ which only depends on the number of ones $|x|$ in the n-bit input x, and not on their locations. To get a feel for Paturi's theorem, consider the special case of a function $g_k : \{0,1\}^n \to \{0,1\}$ where $g_k(x) = 0$ unless $|x| = k$ in which case $g_k(x) = 1$. Paturi's theorem says that the $\frac{1}{4}$-error approximate degree of g_k, denoted by $\deg_{1/4}(g_k)$, is $\Theta(\sqrt{k(n-k)})$. Later, the ℓ_∞-approximate degree of symmetric boolean functions was characterized for all approximation errors ε by [28, 34]. Again in the special case of g_k, these results say that the degree of a polynomial that approximates g_k up to error $\varepsilon \geq 2^{-n}$ is $\Theta(\sqrt{k(n-k)} + \sqrt{n \log(1/\varepsilon)})$.

1.2 Our results on sum-of-squares approximation

Here we study the representation of non-negative functions on the boolean hypercube by *sums of squares* of polynomials. More precisely, a non-negative boolean function $f : \{0,1\}^n \to \mathbb{R}_+$ has an (exact) *degree-d sum-of-squares (sos) representation* if there exist degree-d polynomials $h_1, \ldots, h_r$ over the reals such that for all $x \in \{0,1\}^n$,

$$f(x) = h_1(x)^2 + \cdots + h_r(x)^2.$$

Let sos-deg(f) be the minimum d such that a non-negative function f has a degree-d sum-of-squares representation.[1] This sos degree is an important quantity that arises in the context of optimization and proof complexity, as also witnessed by our applications below.

The obvious fact that a sum of squares of polynomials is globally non-negative is remarkably useful. For example, for a graph $G = ([n], E)$, if $f(x_1, \ldots, x_n) = c - \sum_{(i,j)\in E}(x_i - x_j)^2$ has an sos representation on the boolean cube, then $c \geq \sum_{(i,j)\in E}(x_i - x_j)^2$ for all $x \in \{0,1\}^n$, and hence G has no cut of size larger than c. Moreover if f has a degree-d sos representation for small d, then this provides a small *certificate* (of size $n^{O(d)}$) that f has no cut of size larger than c. Such certificates can in fact be found by means of semidefinite programming; these observations are the basis of the semidefinite programming hierarchies of Lasserre and Parrilo [31, 18, 24] that have been the subject of intense study in approximation algorithms.

While exact sum-of-squares degree of functions on the boolean hypercube has been previously studied, there has been little work on the *approximation* of such functions by sos polynomials. This is the focus of our paper, and we prove a number of tight bounds on the

[1] Note that the degree of the polynomial representing f will actually be $2d$.

approximate sum-of-squares degree of functions on the hypercube. We consider two notions of approximation in this paper. The most familiar is ℓ_∞-approximation: an sos polynomial h ε-approximates a function $f : \{0,1\}^n \to \mathbb{R}_+$ in ℓ_∞-distance if $|f(x) - h(x)| \leq \varepsilon$ for all $x \in \{0,1\}^n$. We let $\text{sos-deg}_\varepsilon(f, \ell_\infty)$ denote the minimum degree of an sos polynomial that ε-approximates f in ℓ_∞-distance. The other notion is ℓ_1-approximation: an sos polynomial h ε-approximates f in ℓ_1-distance if $\sum_{x\in\{0,1\}^n} |f(x) - h(x)| \leq \varepsilon$, and we let $\text{sos-deg}_\varepsilon(f, \ell_1)$ denote the minimum degree of an sos polynomial that ε-approximates f in ℓ_1-distance. Note that $\varepsilon = \delta 2^n$ corresponds to *average* approximation error δ.[2]

For much of this paper we will focus on understanding the approximate sos degree of the symmetric quadratic functions $f_k : \{0,1\}^n \to \mathbb{R}_+$ defined as $f_k(x) = (|x| - k)(|x| - k - 1)$ for $k = 0, 1, 2, \ldots, n-1$. Our study of these functions is motivated by several reasons. First, these functions have a close connection to the proof complexity of the knapsack problem [12], and have recently been used to show lower bounds on semidefinite extension complexity [19]; we survey these two applications in Section 1.3 below. Furthermore, while the f_k may look very special, they are not too far from a general symmetric quadratic polynomial with real coefficients that is nonnegative on the hypercube, for the following reason. Any symmetric quadratic polynomial on the hypercube is of the form $p(|x|)$ for a quadratic univariate polynomial p. The polynomial p will have two roots. If the roots are complex they must come in a conjugate pair and p is already sos; if the roots are real, and not both ≤ 0 or $\geq n$, then they must lie in an interval $[k, k+1]$, for some $k \in \{0, 1, \ldots, n\}$, just as with f_k.

In our first set of results, we give lower and upper bounds on the ℓ_∞-approximate sos degree of the functions f_k.

▶ **Theorem 1.1** (ℓ_∞ sos approximations of f_k)**.** *For all integers $n \geq 0$, $k \in \{1, \ldots, n-2\}$, and $\varepsilon = \varepsilon(n)$ satisfying $0 < \varepsilon < 1/50$, we have*

1. $\textit{sos-deg}_\varepsilon(f_k, \ell_\infty) = \Omega(\sqrt{k(n-k)})$
2. $\textit{sos-deg}_\varepsilon(f_k, \ell_\infty) = O(\sqrt{k(n-k)} + \sqrt{n \log(1/\varepsilon)})$

We expect the lower bound can be improved for the case of small ε to match the upper bound, but have been unable to show that so far. Observe that in the case of constant error, we obtain the tight bound of $\text{sos-deg}_{1/50}(f_k, \ell_\infty) = \Theta\left(\sqrt{k(n-k)}\right)$. While we are not aware of any previous work on ℓ_∞-approximate sos degree, techniques of Grigoriev [12] can be used to show that, for n odd, any degree-$(n-1)/2$ sos polynomial has error at least $\Omega(1/\log n)$ for approximating $f_{\lfloor n/2 \rfloor}$. This derivation is given in Appendix D.

The similarity between our ℓ_∞ bounds for f_k and Paturi's bound for the 0/1-valued functions g_k defined above is striking. For the upper bound, the connection can be seen as

[2] Also note that the existence of a degree-d sos approximation in either of these notions can be formulated as the feasibility of a semidefinite program of size polynomial in the domain size 2^n, as follows. For $x \in \{0,1\}^n$, let m_x be the column vector of dimension $D = \sum_{i=0}^{d} \binom{n}{i}$ indexed by sets $S \subseteq [n]$ of size $\leq d$, with entry $m_{x,S} = \prod_{i\in S} x_i$. Let M_x be the $D \times D$ rank-1 matrix $m_x m_x^T$. Suppose $p(x) = \sum_{S:|S|\leq d} p_S \prod_{i\in S} x_i$ is a multilinear polynomial of degree d, where with slight abuse of notation we use p also to denote the D-dimensional vector of real coefficients p_S. Then $p(x)$ is the inner product of p and m_x, so $p(x)^2 = \text{Tr}(pp^T M_x)$. Accordingly, every sos polynomial h of degree $\leq d$ corresponds to a psd matrix Z such that $h(x) = \text{Tr}(ZM_x)$ for all $x \in \{0,1\}^n$ (Z can be written as $\sum_{i=1}^{r} p_i p_i^T$, so the rank r of Z would be the number of squared polynomials that h sums over). Hence the existence of an sos polynomial h of degree $\leq d$ that ε-approximates f in ℓ_∞-distance, is equivalent to the existence of a psd matrix Z such that $|\text{Tr}(ZM_x) - f(x)| \leq \varepsilon$ for all $x \in \{0,1\}^n$. The latter corresponds to the feasibility of an SDP with 2^n constraints, which (up to issues of precision) can be solved in time $2^{O(n)}$. However, we won't use this fact in this paper.

follows: we can construct an ε-approximation to f_k in ℓ_∞-distance by finding a univariate polynomial $e(z)$ such that $h(z) = (z-k)(z-k-1) + e(z)$ is globally nonnegative (i.e., $h(z) \geq 0$ for all $z \in \mathbb{R}$), and $|e(i)| \leq \varepsilon$ on integer points $i \in \{0, 1, \ldots, n\}$. As $h(z)$ is a globally nonnegative *univariate* polynomial, it is sos, and furthermore $h(|x|)$ is an ε-approximation to f_k. What are the requirements on e? It must be large enough to "cancel out" the negative values of $(z-k)(z-k-1)$ in the interval $(k, k+1)$, but small on all integer points $0, 1, \ldots, n$. This is very similar to looking for an ε-approximation of g_k, and techniques similar to those used by Paturi show that there is an e satisfying these requirements of degree $O(\sqrt{k(n-k)} + \sqrt{n}\log(1/\varepsilon))$. Note that this is slightly weaker than what Theorem 1.1 claims; we will soon discuss how to bring the $\log(1/\varepsilon)$ inside the square-root.

This upper bound argument shows that f_k can be approximated by a polynomial $h(|x|)$ where h is globally nonnegative. For the lower bound, it is not clear at all why the optimal approximating polynomial should be of this form. Any symmetric polynomial $f(x_1, \ldots, x_n)$ on the hypercube is of the form $f(x_1, \ldots, x_n) = p(|x|)$ for a univariate polynomial p. Even if f is sos, however, this does not mean that p will be globally nonnegative.

For the lower bound, we use an elegant recent result of Blekherman [2] that gives a characterization of the possible form of univariate polynomials p such that $f(x_1, \ldots, x_n) = p(|x|)$ when f is a sos and symmetric real-valued boolean function. This structural theorem allows us to reduce the analysis of the approximate sos degree of f_k to the approximate degree of a symmetric function on the boolean hypercube, for which we can apply Paturi's lower bound.

Interestingly, for small ε we can show a better upper bound than that given by the argument sketched above. To get the better upper bound of Theorem 1.1, we take advantage of a recent characterization of the sos degree of a non-negative real-valued boolean function as the quantum query complexity of computing that function *in expectation* (see Section 1.3.3). We explicitly design a quantum algorithm to approximately compute f_k in expectation with query complexity $O(\sqrt{k(n-k)} + \sqrt{n\log(1/\varepsilon)})$, which by the characterization implies the same upper bound on $\text{sos-deg}_\varepsilon(f_k, \ell_\infty)$. This again parallels the situation for symmetric boolean-valued functions, where the tight upper bound of $O(\sqrt{k(n-k)} + \sqrt{n\log(1/\varepsilon)})$ on $\deg_\varepsilon(g_k)$ was first shown by the construction of a quantum query algorithm [34].

We also study sos ℓ_1-approximations of f_k:

▶ **Theorem 1.2** (sos ℓ_1-approximations of f_k)**.** *Let n be odd and $k = \lfloor n/2 \rfloor$. Then*

$$\textit{sos-deg}_{\delta 2^n}(f_k, \ell_1) \leq \left\lceil \frac{3\sqrt{n}}{\sqrt{2}\delta} \ln\left(\frac{1}{\delta}\right) \right\rceil ,$$

for any $8/\sqrt{2n} \leq \delta \leq 1/4$. For $k < 0.49n$, we have $\textit{sos-deg}_{\delta 2^n}(f_k, \ell_1) = O\left(\ln\left(\frac{1}{\delta}\right)\right)$.

The proof of this theorem follows the same plan sketched above for the upper bound in the ℓ_∞ case. We construct a low-degree univariate polynomial $e(z)$ such that $h(z) = (z-k)(z-k-1) + e(z)$ is globally nonnegative and $\varepsilon = \sum_{i=0}^{n} \binom{n}{i} |e(i)|$ is relatively small. Then $h(|x|)$ gives the desired sos approximation to f_k in ℓ_1-norm. We discuss the applications of this theorem to the lower bounds on semidefinite extension complexity of [19] below in Section 1.3.1.

1.3 Applications in complexity theory

Here we describe complexity-theoretic consequences of such sos bounds in three different settings.

1.3.1 Positive semidefinite extension complexity

The approximation of boolean functions by sos polynomials has played an important role in *inapproximability* results. Our first application is to the analysis of the positive semidefinite extension complexity of polytopes. The recent breakthrough work of Lee, Raghavendra and Steurer [19] showed that any semidefinite program whose feasible region projects to the *correlation polytope* must have size $2^{\tilde{\Omega}(n^{2/11})}$. By reduction this in turn implies a $2^{\tilde{\Omega}(n^{1/11})}$ lower bound for the polytope corresponding to the Traveling Salesman Problem on n-vertex graphs, showing (roughly speaking) that TSP cannot be solved by small semidefinite programs.

The argument of [19] shows that lower bounds on the degree of sos polynomials that approximate a function $f_k(x) = (|x|-k)(|x|-k-1)$ in ℓ_1-distance on the boolean cube imply lower bounds on the semidefinite extension complexity of the correlation polytope. They build on the work of Grigoriev [12] to show that, for odd n and $k = \lfloor n/2 \rfloor$, any sum of squares of degree-$\lfloor n/2 \rfloor$ polynomials has ℓ_1-error at least $2^{n-2}/\sqrt{n}$ in approximating $f_{\lfloor n/2 \rfloor}$.[3] Our Theorem 1.2 shows that this bound is tight, up to logarithmic factors. Further, our upper bound on $\text{sos-deg}_{\delta 2^n}(f_k, \ell_1)$ throughout the full range of error implies, roughly speaking, that the quantitative bounds of [19] cannot be improved simply by showing better sos degree lower bounds on f_k.

1.3.2 Proof complexity

Our second result is in proof complexity. Grigoriev and Vorobjov [14] introduced a proof system based on the Positivstellensatz [32]. We explain this proof system in the context of the knapsack problem. In this instance, the knapsack problem can be phrased as looking for a solution $x \in \mathbb{R}^n$ to the system of equations

$$\sum_{i=1}^{n} x_i - r = 0,\ x_1^2 - x_1 = 0, \ldots, x_n^2 - x_n = 0 \tag{1}$$

where $r \in \mathbb{R}$. When r is not an integer, this system obviously has no solution. One way to certify that there is no solution, is to find polynomials $g, g_1, \ldots, g_n$ and sos polynomial h such that

$$g(x) \cdot \left(\sum_{i=1}^{n} x_i - r \right) + \sum_{i=1}^{n} g_i(x) \cdot (x_i^2 - x_i) = 1 + h(x)\,. \tag{2}$$

Such a collection of polynomials constitutes a *Positivstellensatz refutation* of the statement that (1) has a solution: if $a \in \mathbb{R}^n$ satisfied $\sum_{i=1}^n a_i - r = 0$, and $a_i^2 - a_i = 0$ for $i = 1, \ldots, n$, then the left-hand side of (2) would evaluate to 0 on a, while the right-hand side would evaluate to $1 + h(a) \geq 1$, a contradiction.

Grigoriev and Vorobjov define the *Positivstellensatz refutation degree* of the system (1) as

$$\max \left\{ \deg \left(g(x) \cdot \left(\sum_{i=1}^{n} x_i - r \right) \right), \max_{i \in [n]} \{ \deg(g_i(x) \cdot (x_i^2 - x_i)) \}, \deg(h(x)) \right\},$$

[3] The initial version of [19] only claimed a lower bound on the ℓ_1-error of $\Omega(2^n/n^{3/2})$. However, their argument actually shows a bound of $\Omega(2^n/\sqrt{n})$ after a computational error is corrected. This improves the bound on the psd extension complexity of the correlation polytope from the $2^{\tilde{\Omega}(n^{2/13})}$, of their paper, to the $2^{\tilde{\Omega}(n^{2/11})}$ quoted here.

maximized over all sets of polynomials satisfying (2). Grigoriev [12] shows that if $k < r < k+1$ for a nonnegative integer $k < (n-3)/2$, then any Positivstellensatz refutation of (1) has degree at least $2k+4$. We provide a simple proof of this in Appendix C using Blekherman's theorem. Kurpisz et al. [17] independently also give an alternative proof of Grigoriev's lower bound, by showing a more general theorem that reduces the analysis of dual certificates for very symmetric sos proof systems (such as for knapsack) to the analysis of univariate polynomials.

Grigoriev's lower bound shows the weakness of the Positivstellensatz-based proof system: even to refute such easy instances it already needs polynomials of fairly high degree. We prove here that Grigoriev's lower bound is exactly tight, answering an open question from [12].

▶ **Theorem 1.3.** *Let $k < r < k+1$ for a nonnegative integer k. The Positivstellensatz refutation degree of* (1) *with this value of r is at most $2k+4$.*

1.3.3 Quantum query complexity of approximating a function in expectation

The third application is to quantum algorithms. Kaniewski et al. [16] observed a very close connection between the sos degree of a function $f : \{0,1\}^n \to \mathbb{R}_+$ and a variant of quantum query complexity: sos-deg(f) is *exactly equal* to the optimal query complexity among all quantum algorithms with non-negative outputs whose *expected* output on input x equals $f(x)$.[4] This model of query complexity in expectation is motivated by similar models of *communication* complexity that arose in the study of extension complexity of polytopes [9].

However, as [16] note, this model has some intrinsic interest and motivation as well. Suppose we want to approximate $F(x) = \sum_{i=1}^m f_i(x)$, where each f_i is a non-negative function of $x \in \{0,1\}^n$. Then we can just compute, for each i, a random variable whose expected value is $f_i(x)$ and then output the sum of those random variables. By linearity of expectation, the output will have the correct expectation $F(x)$. It will be tightly concentrated around its expectation if the individual random variables have a variance that is not too large. Thus in some cases it suffices to compute the $f_i(x)$ in expectation only, rather than to compute the values $f_i(x)$ themselves (which may be much more expensive). In this example, it is actually not even necessary to compute each $f_i(x)$ *exactly* in expectation. If the ith random variable has an expectation that is within ε_i of $f_i(x)$, then the expected value of our output is within $\sum_{i=1}^m \varepsilon_i$ of the correct value $F(x)$.

The same proofs that Kaniewski et al. [16] used to equate sos-deg(f) and quantum query complexity in expectation also work in the approximate case. For example, sos-deg$_\varepsilon(f, \ell_\infty)$ is the optimal query complexity among all quantum algorithms with non-negative outputs whose expected output on input x differs from $f(x)$ by at most ε, for every $x \in \{0,1\}^n$, and the analogous statements hold for approximation using the other norms. Accordingly, our above results about approximate sos degree immediately translate to results about quantum query complexity of algorithms that approximate f in expectation.

1.4 Organization

The rest of the paper is organized as follows. In Section 2, we prove our sos ℓ_∞-approximation bounds (Theorem 1.1). In Section 3 we prove upper bounds on the degree of sos ℓ_1-ap-

[4] To avoid potential confusion: for each fixed x the expectation is taken over the internal randomness of the algorithm; it is not an expectation over different inputs x.

proximations to $f_{\lfloor n/2 \rfloor}$ (Theorem 1.2), and show that the lower bound of [19] on the extension complexity of the correlation polytope cannot be improved by obtaining better sos ℓ_1-approximate degree lower bounds. In Section 4 we prove Theorem 1.3, showing tightness of Grigoriev's knapsack lower bound.

2 Sum-of-squares approximation in ℓ_∞-norm

In this section we give lower and upper bounds on the ℓ_∞-approximate sos degree of the function $f_k(x) = (|x| - k)(|x| - k - 1)$ to prove Theorem 1.1.

▶ Remark. Throughout this section, we will assume that $k \leq n/2$. Letting $\bar{x}$ denote the bitwise complement of x, we see that $f_k(x) = (|\bar{x}|-(n-k-1))(|\bar{x}|-(n-k)) = (|\bar{x}|-\ell)(|\bar{x}|-\ell-1)$ where $\ell = n-k-1$. Thus if $k > n/2$, then $\ell \leq n/2$ and $f_k(x) = f_\ell(\bar{x})$. For any $h : \{0,1\}^n \to \mathbb{R}$ the functions $h(x)$ and $h(\bar{x})$ have the same sos degree, so it suffices to work with f_k for $k \leq n/2$.

2.1 Lower bound preliminaries

The following lemma is implicit in Paturi [25].

▶ **Lemma 2.1** (Paturi [25]). *Let $p : \mathbb{R} \to \mathbb{R}$ be a univariate polynomial and suppose that $0 \leq p(i) \leq c$ for all $i \in \{0, 1, \ldots, n\}$. If $|p'(\alpha)| \geq \delta$ for some $0 \leq \alpha \leq n$, then* $\deg(p) = \Omega(\frac{\delta}{c}\sqrt{\alpha(n-\alpha)})$.

We give a simple, but convenient, application of this lemma to the case where p is bounded on $\{0, 1, \ldots, n\}$, except possibly for a small set S near where p is known to be small.

▶ **Lemma 2.2.** *Let $p : \mathbb{R} \to \mathbb{R}$ be a univariate polynomial, $S \subseteq \{0, 1, \ldots, n\}$, and suppose that the following bounds are known.*

- *$|p(i)| \leq c$ for all $i \in \{0, 1, \ldots, n\} \setminus S$, for a constant c.*
- *$p(\alpha) \leq \varepsilon$, for some $\alpha \in \{0, 1, \ldots, \lfloor n/2 \rfloor\}$.*
- *$p(\beta) \geq a$ where $|\alpha - \beta| \leq d_1$, for a constant d_1.*
- *$\max_{i \in S} |i - \alpha| \leq d_2$, for a constant d_2.*

If $a \geq c > \varepsilon$, then $\deg(p) = \Omega(\sqrt{\alpha(n-\alpha)})$, where the constant in the $\Omega(.)$ depends on c, d_1, d_2.

Proof. If $|p(i)| \leq c$ for all $i \in S$, then applying Paturi's lemma directly we obtain a bound of

$$\Omega\left(\frac{a-\varepsilon}{d_1 c}\sqrt{(\alpha - d_1)(n - \alpha + d_1)}\right).$$

Otherwise, suppose the maximum of $|p(i)|$ over $i \in \{0, 1, \ldots, n\}$ is attained at $j \in S$. Then the derivative of p is at least $(|p(j)| - \varepsilon)/d_2$. Applying Paturi's lemma in this case gives a bound of

$$\Omega\left(\frac{|p(j)| - \varepsilon}{d_2 |p(j)|}\sqrt{(\alpha - d_2)(n - \alpha + d_2)}\right) \geq \Omega\left(\frac{1}{d_2}\left(1 - \frac{\varepsilon}{c}\right)\sqrt{(\alpha - d_2)(n - \alpha + d_2)}\right). \quad \blacktriangleleft$$

We will also need the following elegant theorem of Blekherman [2]. Recall that a *symmetric* real-valued boolean function $f : \{0,1\}^n \to \mathbb{R}$ satisfies $f(x) = f(\pi(x))$ for all $x \in \{0,1\}^n$ and $\pi \in S_n$, where the permutation π acts as $\pi((x_1, \ldots, x_n)) = (x_{\pi(1)}, \ldots, x_{\pi(n)})$. For any symmetric boolean function f of degree d, there is a univariate polynomial $\tilde{f}$ of degree d such that $f(x_1, \ldots, x_n) = \tilde{f}(x_1 + \cdots + x_n)$.

▶ **Theorem 2.3** (Blekherman [2]). *Let $f : \{0,1\}^n \to \mathbb{R}_+$ be a symmetric non-negative real-valued boolean function and $\tilde{f}$ a univariate polynomial such that $f(x_1, \ldots, x_n) = \tilde{f}(x_1 + \cdots + x_n)$. If f can be written as the sum of squares of n-variate polynomials of degree $d \leq n/2$, then we can write*

$$\begin{aligned}\tilde{f}(z) = q_d(z) + z(n-z)q_{d-1}(z) + z(z-1)(n-z)(n-1-z)q_{d-2}(z) + \cdots \\ \cdots + z(z-1)\cdots(z-d+1)(n-z)(n-1-z)\cdots(n-d+1-z)q_0(z)\end{aligned} \tag{3}$$

where each $q_t(z)$ is a univariate sos polynomial with sos-deg$(q_t) \leq t$.

In Appendix B we include a proof of Blekherman's theorem. In Appendix C, we use Blekherman's theorem to provide a simple proof of Grigoriev's lower bound [12] on the degree of Positivstellensatz refutations for the knapsack problem.

2.2 Lower bound for exact sos degree

To illustrate our proof technique, we first show how the above tools can be used to prove a bound of $\Omega(\sqrt{k(n-k)})$ on the exact sos degree of $f(x) = (|x| - k)(|x| - k - 1)$. In the next section, we will extend this proof to also work for the approximate case.

▶ **Theorem 2.4.** *Let $f_k : \{0,1\}^n \to \mathbb{R}_+$ be defined as $f_k(x) = (|x| - k)(|x| - k - 1)$ for an integer $1 \leq k \leq n-2$. Then sos-deg$(f_k) = \Omega(\sqrt{k(n-k)})$.*

Proof. Following Remark 2, if we show the theorem for $1 \leq k \leq n/2$, then it will also imply the theorem for $1 \leq k \leq n-2$. We thus assume $1 \leq k \leq n/2$.

Let d be the sos degree of f. We may assume $d \leq n/2$ as otherwise the theorem holds. Let $\tilde{f}$ be a univariate polynomial of degree $\leq 2d$ such that $\tilde{f}(x_1 + \cdots + x_n) = f(x_1, \ldots, x_n)$. Write $\tilde{f}(z) = g_1(z) + g_2(z)$ where $g_1(z) = q_d(z) + z(n-z)q_{d-1}(z) + \cdots + z(z-1)\cdots(z-(k-1))(n-z)(n-1-z)(n-(k-1)-z)q_{d-k}(z)$ is the first $k+1$ terms in the representation of $\tilde{f}$ of Theorem 2.3, and $g_2(z)$ is the remaining part of that representation.

Our first claim is that $(z-k)$ is a factor of both g_1 and g_2. Notice that $\tilde{f}(k) = g_1(k) + g_2(k) = 0$. Furthermore each term of g_1 and g_2 is nonnegative on integer points between 0 and n, which means that each individual term of g_1 and g_2 must evaluate to 0 at k.

Consider now a general term $z(z-1)\cdots(z-t)(n-z)(n-1-z)(n-t-z)q_{d-t-1}(z)$ of Blekherman's representation. If $t \geq k$ then this term obviously has a factor of $z-k$. If $t < k$ then the prefactor $z(z-1)\cdots(z-t)(n-z)(n-1-z)(n-t-z)$ is non-zero for $z = k$, so it must be the case that $q_{d-t-1}(k) = 0$. Since $q_{d-t-1}(z)$ is a univariate sum-of-squares polynomial, even $(z-k)^2$ divides $q_{d-t-1}(z)$.

By the choice of the breakpoint between g_1 and g_2, this shows that $(z-k)^2$ is a factor of g_1 and $z-k$ is a factor of g_2. By the same argument, $(z-(k+1))^2$ is also a factor of g_1, and $(z-(k+1))$ is a factor of g_2.

In light of this, we can write $g_1(z) = (z-k)(z-k-1)h_1(z)$, $g_2(z) = (z-k)(z-k-1)h_2(z)$ so that

$$(z-k)(z-k-1) = \tilde{f}(z) = (z-k)(z-k-1)(h_1(z) + h_2(z))\,.$$

This means that $h_1(i) + h_2(i) = 1$ for all $i \in \{0, 1, \ldots, n\} \setminus \{k, k+1\}$. Furthermore, $h_1(k) = h_1(k+1) = 0$ as h_1 still has roots at $k, k+1$ (as g_1 had double roots there), and $h_2(i) = 0$ for $i \in \{0, \ldots, k-1\}$ because each term in h_2 includes the prefactor $z(z-1)\cdots(z-k+1)$. Combining these observations with the fact that $h_1(i) \geq 0, h_2(i) \geq 0$ for all $i \in \{0, 1, \ldots, n\} \setminus \{k, k+1\}$ gives the following:

1. $0 \leq h_1(i) \leq 1$ for all $i \in \{0, 1, \ldots, n\}$.
2. $h_1(i) = 1$ for $i \in \{0, \ldots, k-1\}$.
3. $h_1(k) = h_1(k+1) = 0$.

Applying Lemma 2.2 to h_1 now gives the desired result. ◀

2.3 Lower bound for ℓ_∞-approximate sos degree

Now we show the lower bound of Theorem 1.1.

▶ **Theorem 2.5.** *Let $f : \{0,1\}^n \to \mathbb{R}_+$ be defined as $f(x) = (|x| - k)(|x| - k - 1)$ for some integer $1 \leq k \leq n-2$. Then sos-deg$_{1/50}(f, \ell_\infty) = \Omega(\sqrt{k(n-k)})$.*

Proof. We now describe how the above proof can be modified to work for ℓ_∞-approximate sum-of-squares degree. We again assume $1 \leq k \leq n/2$. Suppose that $h : \{0,1\}^n \to \mathbb{R}$ is a sum of squares of degree-$d \leq n/2$ polynomials that satisfies $|h(x) - f(x)| \leq \varepsilon$ for all $x \in \{0,1\}^n$, for some $\varepsilon < 1/4$ to be determined later. Let $\tilde{h}$ be the univariate polynomial such that $\tilde{h}(x_1 + \cdots + x_n) = h(x_1, \ldots, x_n)$. Note that $\tilde{h}$ satisfies $|\tilde{h}(i) - (i-k)(i-k-1)| \leq \varepsilon$ for all $i \in \{0, 1, \ldots, n\}$. We again use Blekherman's theorem to decompose $\tilde{h}(z) = g_1(z) + g_2(z)$ where, as before, $g_1(z) = q_d(z) + z(n-z)q_{d-1}(z) + \cdots + z(z-1)\cdots(z-k+1)(n-z)(n-1-z)(n-k+1-z)q_{d-k}(z)$. The polynomial g_1 has the following properties.

1. $g_1(i) = \tilde{h}(i)$ for $i \in \{0, 1, \ldots, k\}$, because all terms of g_2 are zero on these points.
2. $g_1(i) \leq \tilde{h}(i)$ for $i \in \{0, 1, \ldots, n\}$. This follows as $g_2(i)$ is nonnegative on integer points in $\{0, 1, \ldots, n\}$.
3. $g_1(i) \geq 0$ for $i \in [k-1, n-k+1]$. Each term of g_1 is nonnegative in this interval because the prefactor is.

We will consider two cases based on the value of $g_1(k + 3/2)$. First consider the case $g_1(k + 3/2) > \varepsilon$. In this case, consider a point $\alpha \in \operatorname{argmin}_z\{g_1(z) : k - 1 \leq z \leq k + 3/2\}$. Let $g_1(\alpha) = \delta$. By item (3) above and as $g_1(k-1), g_1(k+3/2) > \varepsilon$ and $g_1(k), g_1(k+1) \leq \varepsilon$ we have $0 \leq \delta \leq \varepsilon$ and also $g_1'(\alpha) = 0$.

Now consider the function $p_1 = g_1 - \delta$. As $p_1(\alpha) = p_1'(\alpha) = 0$ it follows that p_1 has a double root at α. Define q_1 by $p_1(z) = (z-\alpha)^2 q_1(z)$. Note that q_1 has the following properties.

1. $q_1(i) \leq 6 + \varepsilon$ for $i \in \{0, 1, \ldots, n\} \setminus \{k-1, k, k+1, k+2\}$.
2. $q_1(k-1) \geq \frac{2-2\varepsilon}{9}$.
3. As either $|\alpha - k| \geq 1/2$ or $|\alpha - k - 1| \geq 1/2$ we have either $q_1(k) \leq 4\varepsilon$ or $q_1(k+1) \leq 4\varepsilon$.

Applying Lemma 2.2 then gives the desired lower bound in this case as long as $\varepsilon < 1/19$.

Now we consider the second case, that $g_1(k + 3/2) \leq \varepsilon$. In this case, we modify g_1 by adding a function that is shaped like a "smile." Let $p_1(z) = g_1(z) + 8\varepsilon(x-k-1)(x-k-2)$. Note that p_1 satisfies $p_1(k+1) \geq 0$, $p_1(k+3/2) \leq -\varepsilon$, and $p_1(k+2) \geq 0$. Thus $p_1(z)$ has two roots α, β in $[k+1, k+2]$, with $\alpha \leq \beta$. Let $p_1(z) = (z-\alpha)(z-\beta)r_1(z)$. Then r_1 satisfies the following properties.

1. $r_1(i) \leq 2$ for $i \in \{0, 1, \ldots, n\} \setminus \{k+1, k+2\}$.
2. $r_1(k-1) \geq \frac{2}{9} + 5\varepsilon$.
3. $|r_1(k)| \leq 16\varepsilon$.

Applying Lemma 2.2 then gives the desired lower bound as long as $\varepsilon < 2/99$. ◀

2.4 Upper bound for ℓ_∞-approximate sos degree

In this section we show that the lower bound in Theorem 2.5 is tight. To do this, we use the characterization of the sos degree of a function $f : \{0,1\}^n \to \mathbb{R}_+$ as the quantum query

complexity of computing f in expectation [16]. In this model, a quantum algorithm A makes a number T of quantum queries to the hidden input x, and outputs a non-negative real number. We say that the algorithm A *computes f in expectation* if the expected value of the output of the algorithm A on input x is exactly equal to $f(x)$. We will use $\mathrm{QE}(f)$ to denote the minimum number of quantum queries T needed by such an algorithm to compute f in expectation. Kaniewski et al. [16] show that $\mathrm{QE}(f)$ exactly captures the sos degree of f:

▶ **Theorem 2.6** ([16]). *Let $f : \{0,1\}^n \to \mathbb{R}_+$. Then $\mathrm{QE}(f) = \textit{sos-deg}(f)$.*

Thus, in order to prove an upper bound on the (approximate) sos degree of a function f, it suffices to construct a quantum query algorithm that (approximately) computes f in expectation. The only knowledge of quantum query complexity needed to understand the algorithm is Theorem 2.6 above, and the existence of the following quantum algorithms, all of which are variants of Grover search.

- *Regular Grover* [15, 4]: If $|x| \geq t$ then there is a quantum algorithm (depending on t) using $O(\sqrt{n/t})$ queries that finds an i such that $x_i = 1$ with probability at least $1/2$.
- *ε-error Grover* [5]: There is a quantum algorithm using $O(\sqrt{n \log(1/\varepsilon)})$ queries that finds an i such that $x_i = 1$ with probability at least $1 - \varepsilon$ if $|x| \geq 1$.
- *Exact Grover* [4]: If $|x| = t$ then there is a quantum algorithm (depending on t) using $O(\sqrt{n/t})$ queries that finds an i such that $x_i = 1$ *with certainty*.

The algorithm consists of three subroutines, which we now describe. We begin with the simplest procedure, $\mathrm{SAMPLE}(x, S)$, which motivates the basic plan of the algorithm.

Algorithm 1 Given $x \in \{0,1\}^n$ and $S \subseteq [n]$, samples two entries of x outside of S

1: **procedure** SAMPLE(x, S)
2: Randomly choose $i \neq j \in [n] \setminus S$. Output $x_i x_j \cdot (n - |S|)(n - |S| - 1)$
3: **end procedure**

▶ **Claim 2.7.** *The procedure SAMPLE(x, S) makes two queries and the expected value of its output is $(|x| - |S|)(|x| - |S| - 1)$.*

The procedure SAMPLE suggests the following high-level idea for an algorithm for computing $f_k(x) = (|x| - k)(|x| - k - 1)$. For simplicity we describe the high-level idea for the case where $k \leq n/2$ and where we want to compute a constant-error ε-approximation of f_k, in expectation.

First we try to find a set S of k ones in x assuming that $|x| > 2k$, using a procedure HIGH. If we find such a set S then we run $\mathrm{SAMPLE}(x, S)$ and output $f(x)$ exactly, in expectation. If the procedure HIGH fails to find such a set S, then we run a procedure LOW. This uses exact Grover search to determine the Hamming weight of x with certainty if $|x| \leq 2k$. Once we know the Hamming weight of x we can correctly output $f(x)$, deterministically. Both the procedures HIGH and LOW can be done with $O(\sqrt{kn})$ queries, in the constant error ε case. The only case where the algorithm may err is if $|x| > 2k$ but the procedure HIGH fails to find k ones in x. The most subtle part of the algorithm is tuning the parameters such that this error is at most ε in expectation. We now describe the procedures HIGH and LOW.

▶ **Lemma 2.8.** *Fix δ and let $M = \max\{k, \lceil \log(1/\delta) \rceil\}$. Suppose that $|x| \geq t > 2k$. Then procedure HIGH(x, t, δ) makes $O(M\sqrt{n/t})$ queries and returns a set S with $|S| = k$ and $x_i = 1$ for all $i \in S$ with probability at least $1 - \delta$.*

Algorithm 2 Find k ones in x with probability $1-\delta$, assuming $|x| \geq t > 2k$

```
procedure HIGH(x, t, δ)
    S = ∅
    ℓ = 1
    while ℓ ≤ 5 max(k, ⌈log(1/δ)⌉) and |S| < k do
        ℓ ← ℓ + 1
        Grover search assuming |x| ≥ t/2.
        if find x_i = 1 then S ← S ∪ i, x ← x \ x_i
        end if
    end while
    return S
end procedure
```

Proof. As each Grover search requires $O(\sqrt{n/t})$ queries, in total the procedure makes $O(M\sqrt{n/t})$ queries. Let us now estimate the probability that it exits without finding a set S of size k.

As we are given that initially $|x| \geq t > 2k$, if less than k ones are found then throughout the algorithm there remain at least $t/2$ ones in x. Thus each run of Grover has probability of success at least $1/2$. The probability to have fewer than k successes among the $5M$ runs is therefore at most

$$\frac{1}{2^{5M}} \sum_{i=0}^{k-1} \binom{5M}{i} \leq 2^{-(1-H(k/5M))5M} \leq 2^{-M} \leq 2^{-\log(1/\delta)} = \delta,$$

where $H(\cdot)$ denotes binary entropy, and we used that $1 - H(k/5M) \geq 1 - H(1/5) \geq 1/5$. ◀

Next we give the algorithm LOW.

Algorithm 3 Outputs $(|x|-k)(|x|-k-1)$ with certainty if $|x| \leq t$

```
procedure LOW(x, t)
    S = ∅
    for i = t to 1 do
        Exact Grover search assuming |x| = i
        if find x_i = 1 then S ← S ∪ i, x ← x \ x_i
        end if
    end for
    Output (|S| − k)(|S| − k − 1).
end procedure
```

▶ **Claim 2.9.** *If $|x| \leq t$, then LOW(x,t) outputs $(|x|-k)(|x|-k-1)$ and makes $O(\sqrt{tn})$ queries.*

Proof. The number of queries is

$$\sum_{i=1}^{t} O(\sqrt{n/i}) = O(\sqrt{tn}).$$

Next we show that if $|x| \leq t$, then LOW(x,t) will find all of the ones in x (this is similar to [11]). Initially the index $i = t$ and thus $i \geq |x|$. This invariant is maintained throughout

the algorithm. If ever $i = |x|$ then we will find all the remaining ones in x as our guess for the number of ones is always correct after this point. On the other hand, if the algorithm terminates with $i = 1 > |x|$ then we have found all the ones in the original input x. ◀

With these procedures in place, we can describe the main algorithm and prove its correctness.

Algorithm 4 Main

```
1: procedure MAIN(x, ε)
2:     m = max(k, ⌈log(1/ε)⌉)
3:     for i = 1 to ⌊log(n/m)⌋ do
4:         t ← 2^i m
5:         δ ← ε/(4t^2)
6:         S=HIGH(x, t, δ)
7:         if |S| = k then
8:             SAMPLE(x, S)
9:             Exit
10:        end if
11:    end for
12:    LOW(x, 2m)
13: end procedure
```

▶ **Theorem 2.10.** *For every $x \in \{0,1\}^n$, the expected value of Main(x, ε) differs from $(|x| - k)(|x| - k - 1)$ by at most ε. The algorithm makes at most $O(\sqrt{kn} + \sqrt{n \log(1/\varepsilon)})$ queries.*

Proof. Following Remark 2 we may assume that $k \leq n/2$. First we verify the stated complexity of the algorithm. Note that by definition of m in the main Algorithm 4, it suffices to show that the algorithm makes $O(\sqrt{nm})$ queries. By Claim 2.9 the call to LOW$(x, 2m)$ makes $O(\sqrt{mn})$ queries, and by Claim 2.7 there are at most 2 queries made by SAMPLE as this is called at most once. Finally, the number of queries in the call to HIGH when $t = 2^i m$ and $\delta = \varepsilon/(4t^2)$ is at most

$$O\left(k\sqrt{\frac{n}{2^i m}} + \log(2^{2i+2}m^2/\varepsilon)\sqrt{\frac{n}{2^i m}}\right) = O\left(\sqrt{\frac{kn}{2^i}} + \sqrt{\frac{n\log(1/\varepsilon)}{2^i}} + \log(2^{2i+2}m^2)\sqrt{\frac{n}{2^i m}}\right)$$

where we have used the fact that $m \geq k$ and $m \geq \log(1/\varepsilon)$. The sum of the first two terms over $i \geq 1$ is $O(\sqrt{kn} + \sqrt{n \log(1/\varepsilon)})$ as desired. As for the sum of the third term, we have

$$\sum_{i \geq 1} O\left(\log(2^{2i+2}m^2)\sqrt{\frac{n}{2^i m}}\right) = O\left(\log(m)\sqrt{\frac{n}{m}}\right) = O(\sqrt{n})\,.$$

We now verify correctness. If $|x| \leq 2m$ then the algorithm will output $(|x| - k)(|x| - k - 1)$ in expectation exactly: if k ones are found in x by a call to HIGH then this will be done by SAMPLE, otherwise all ones in x will be found with certainty by LOW, which will then output correctly. If $|x| > 2m$ and a call to HIGH succeeds in finding k ones in x, the algorithm will also output $(|x| - k)(|x| - k - 1)$ exactly, in expectation. Let p be the probability that this does not happen, i.e., that the output on x is given by the procedure LOW. Then the expected value of the output on x is

$$(1 - p)(|x| - k)(|x| - k - 1) + p \cdot \mathbb{E}[\text{LOW}(x, 2m)]\,,$$

and the deviation from the desired output $(|x|-k)(|x|-k-1)$ is

$$p \cdot (\mathbb{E}[\mathrm{LOW}(x,2m)] - (|x|-k)(|x|-k-1)) .$$

Now $\mathrm{LOW}(x,2m)$ will always output a value $0 \leq (\ell-k)(\ell-k-1)$ for some $\ell \in [2m]$, which is always at most the correct value $(|x|-k)(|x|-k-1)$ as $|x| \geq 2m > k$. Therefore the largest difference between these is when $\mathrm{LOW}(x,2m)$ outputs 0, giving

$$|p \cdot (\mathbb{E}[\mathrm{LOW}(x,2m)] - (|x|-k)(|x|-k-1))| \leq p \cdot (|x|-k)(|x|-k-1) \leq p \cdot |x|^2 .$$

We now finally upper bound this error by giving an upper bound on p. Let i and $t = 2^i m$ be such that $t \leq |x| < 2t$. For this value of t and $\delta = \varepsilon/(4t^2)$ the call to $\mathrm{HIGH}(x,t,\delta)$ fails to find a set S of size k with probability at most $\delta \leq \varepsilon/|x|^2$. Thus $p \cdot |x|^2 \leq \delta \cdot |x|^2 \leq \varepsilon$, as desired. ◀

By Theorem 2.6, the characterization of sos degree in terms of quantum query complexity in expectation (Theorem 2.10) gives the upper bound in Theorem 1.1

3 Sum-of-squares approximation in ℓ_1-norm

In this section, we show upper bounds on the sos degree of polynomials to approximate f_k in ℓ_1-norm. In this section we focus on the case where k is $\lfloor n/2 \rfloor$. When $k < 0.49n$ the function f_k is quite easy to approximate in ℓ_1-norm: there is an sos polynomial of degree $O(\ln(1/\delta))$ which gives a $\delta 2^n$-approximation. We omit the details. [5] Our main result on ℓ_1-approximation is the following.

▶ **Theorem 3.1.** *Let n be odd and $k = \lfloor n/2 \rfloor$. Then for any $8/\sqrt{2n} \leq \delta \leq 1/4$*

$$\textit{sos-deg}_{\delta 2^n}(f_k, \ell_1) \leq \left\lceil \frac{3\sqrt{n}}{\sqrt{2}\delta} \ln\left(\frac{1}{\delta}\right) \right\rceil .$$

Lee, Raghavendra, and Steurer [19], building on work of Grigoriev [12], show that in this case $\text{sos-deg}_{2^n/\sqrt{n}}(f, \ell_1) \geq (n-1)/2$. This lower bound was then plugged into their general theorem to lift ℓ_1-approximate sos degree lower bounds to lower bounds on semidefinite extension complexity. By taking $\delta = 3\ln(n)/\sqrt{2n}$, Theorem 3.1 shows that this lower bound on the ℓ_1-error is tight, up to a logarithmic factor. Also, taking δ to be a small additive constant shows that there is a degree-$O(\sqrt{n})$ sos polynomial which, on average, disagrees with f_k by only a small constant. Taken as a whole, Theorem 3.1 implies that the quantitative bounds on the semidefinite extension complexity of the correlation polytope of [19] cannot be improved simply by improving the sos degree lower bounds on the f_k. We now describe the connection to [19] in greater detail.

3.1 The theorem of Lee, Raghavendra, and Steurer

For a function $f : \{0,1\}^n \to \mathbb{R}$ and an integer $N \geq n$, let $M_N^f : \binom{N}{n} \times \{0,1\}^N \to [0,1]$ be the matrix where $M_N^f(S,x) = f(x|_S)$. The *pattern matrix* of f, introduced in the work of

[5] One way to construct such an sos polynomial is to construct a polynomial e as mentioned after Theorem 1.2 from a classical sampling algorithm: query $O(\ln(1/\delta))$ randomly chosen input bits; output some large number if the observed ratio of 1s is very close to k/n, output 0 otherwise. This induces an sos polynomial with the right properties.

Sherstov [29], is a submatrix of M_N^f. The main theorem of Lee, Raghavendra, and Steurer is the following statement. Here a *degree-d pseudo-density* D is a function $D : \{0,1\}^n \to \mathbb{R}$ such that $\mathbb{E}_x[D(x)] = 1$ and $\mathbb{E}_x[D(x)g(x)^2] \geq 0$ for all polynomials g of degree at most $d/2$ on the boolean cube, with the expectation over a uniformly random $x \in \{0,1\}^n$. We use $\|D\|_\infty$ to denote $\max_{x\in\{0,1\}^n} |D(x)|$.

▶ **Theorem 3.2** ([19]). *Let $f : \{0,1\}^n \to [0,1]$. If there exists an $\varepsilon \in (0,1]$ and a degree-d pseudo-density $D : \{0,1\}^n \to \mathbb{R}$ satisfying $\mathbb{E}_x[D(x)f(x)] < -\varepsilon$, then for every $N \geq 2n$*

$$\mathrm{rk}_{\mathrm{psd}}(M_N^f) \geq \left(\frac{c\varepsilon N}{dn^2\|D\|_\infty \log n}\right)^{d/4} \left(\frac{\varepsilon}{\|D\|_\infty}\right)^{3/2} \sqrt{\mathbb{E}_x f(x)}\,,$$

where $c > 0$ is a universal constant.

We do not formally define here the positive semidefinite (psd) rank of a matrix (denoted by $\mathrm{rk}_{\mathrm{psd}}$ above), but remark that psd rank lower bounds are equivalent to semidefinite extension complexity lower bounds. Lee, Raghavendra, and Steurer proved Theorem 3.2 en route to their breakthrough result on superpolynomial size lower bounds for semidefinite programming relaxations of hard optimization problems.

The more pertinent aspect of Theorem 3.2 to us is the role of the degree-d pseudo-density D. Note that, once we fix the degree of the pseudo-density, the bound only depends on the ratio $\mathbb{E}_x[D(x)f(x)]/\|D\|_\infty$. The largest such ratio that a degree-d pseudo-density can achieve is closely related to the best ℓ_1-approximation of f by degree-$d/2$ sos polynomials. The following claim follows from strong duality of semidefinite programming.

▶ **Claim 3.3.** *Let $f : \{0,1\}^n \to \mathbb{R}$. Then sos-deg$_{\delta 2^n}(f,\ell_1) > d$ if and only if there exists a "witness" function $\psi : \{0,1\}^n \to \mathbb{R}$ satisfying $\mathbb{E}_x[f(x)\psi(x)] > \delta$, and $\mathbb{E}_x[p^2(x)\psi(x)] \leq 0$ for all polynomials p of degree at most d, and $\|\psi\|_\infty = 1$.*

For n odd and $k = \lfloor n/2 \rfloor$ Lee et al. [19], building on work of Grigoriev [12], show that there is a degree-$(n-1)$ pseudo-density D such that $\mathbb{E}_x[D(x)f_{\lfloor n/2\rfloor}]/\|D\|_\infty < -\frac{1}{4\sqrt{n}}$. Plugging $f = f_{\lfloor n/2\rfloor}/n^2$ (with this normalization the range is in $[0,1]$) into Theorem 3.2 gives a lower bound of $2^{\widetilde{\Omega}(N^{2/11})}$ on the psd rank of M_N^f for $N = \widetilde{O}(n^{11/2})$. As M_N^f is a submatrix of the slack matrix of the correlation polytope, this gives the desired lower bound on the semidefinite extension complexity of the correlation polytope.

In light of Claim 3.3, if sos-deg$_{\delta 2^n}(f,\ell_1) \leq d$, then there can be no degree-$2d$ pseudo-density with $\mathbb{E}_x[D(x)f(x)]/\|D\|_\infty < -\delta$. The ℓ_1-approximate sos degree upper bounds of Theorem 3.1 therefore imply the non-existence of pseudo-densities with good properties for Theorem 3.2. It can be verified that $2^{\widetilde{\Omega}(N^{2/11})}$ is in fact the best quantitative bound that Theorem 3.2 can show on $\mathrm{rk}_{\mathrm{psd}}(M_N^{f_k})$ over all the functions f_k and tradeoffs between δ and sos-deg$_{\delta 2^n}(f_k,\ell_1)$.

3.2 Proof of Theorem 3.1

Throughout this proof we set $f = f_{\lfloor n/2\rfloor}$. The main idea of the proof of Theorem 3.1 is to construct a univariate polynomial p such that $h(z) = (z - \lfloor n/2\rfloor)(z - \lceil n/2\rceil) + p(z)$ is globally nonnegative (and therefore sos) and $\sum_{i=0}^n \binom{n}{i}|p(i)|$, which is the ℓ_1-error of $h(|x|)$ in approximating f, is reasonably small. We will construct p using Chebyshev polynomials. Similar constructions to what we need have been done before, see for example [28]; as our requirements are somewhat specific, however, we do the construction from scratch.

Let T_d be the Chebyshev polynomial of degree d. We first recall some basic facts about Chebyshev polynomials [27].

▶ **Fact 3.4.** *Let $T_d(z)$ be the Chebyshev polynomial of degree d. Then*

1. $|T_d(z)| \leq 1$ *for* $z \in [-1,1]$.
2. $T_d(z) = \frac{1}{2}\left((z-\sqrt{z^2-1})^d + (z+\sqrt{z^2-1})^d\right)$.
3. $T_{d+1}(z) = 2zT_d(z) - T_{d-1}(z)$.
4. $T_d(z)$ *is monotonically increasing for* $z \geq 1$, *and if* d *is even* $T_d(z) \geq 1$ *for* $z \in \mathbb{R} \setminus [-1,1]$.

▶ **Theorem 3.5.** *Let $n \geq 1$ be an integer, $\varepsilon \in (0, 1/4]$ an error parameter, and let $a \in \mathbb{R}$ satisfy $1/\sqrt{2} \leq a \leq \sqrt{n}/8$. There is a polynomial p of degree at most*

$$\left\lceil \frac{3n}{4\sqrt{2}\,a} \ln\left(\frac{1}{2\varepsilon}\right) \right\rceil + 1$$

with the following properties:

1. $p(z) \geq \frac{1}{4} - z^2$ *for all* z.
2. $|p(z)| \leq \varepsilon$ *for* $z \in [-\frac{n}{2}, -a] \cup [a, \frac{n}{2}]$.
3. $|p(z)| \leq 2$ *for* $z \in [-\frac{n}{2}, -\frac{1}{2}] \cup [\frac{1}{2}, \frac{n}{2}]$.

Proof. Note that we require in particular that $p(0) \geq 1/4$. Roughly speaking, p should have a 'peak' around 0 and then quickly calm down and be bounded on either side of this peak once $|z| \geq a$. The difficulty in constructing p is that its peak is *in between* the intervals on which is bounded. To get around this, we note that is suffices to let $p(z) = \varepsilon q(z^2)$, where q has the properties

1. $q(z) \geq \frac{1}{4\varepsilon} - \frac{z}{\varepsilon}$ for $z \geq 0$.
2. $|q(z)| \leq 1$ for $z \in [a^2, \frac{n^2}{4}]$.
3. $|q(z)| \leq \frac{2}{\varepsilon}$ for $z \in [\frac{1}{4}, \frac{n^2}{4}]$.

Now $q(z)$ is ripe for a construction with Chebyshev polynomials, and this is what we do. For notational convenience, let $L = n/2$. Define the mapping $s(z) = -2(z-a^2)/(L^2-a^2)+1$ that takes the interval $[a^2, L^2]$ to $[-1,1]$. Note that this mapping takes L^2 to -1 and a^2 to 1. Let T_d be the Chebyshev polynomial of *even* degree d (to be chosen later) and define

$$q(z) = T_d(s(z))\,.$$

As $|T_d(z)| \leq 1$ for $z \in [-1,1]$ by Fact 3.4 item (1), it follows that $q(z)$ satisfies condition (2).

We now turn to item (1) and handle the easy cases first. For $z \geq \frac{1}{4}$ we have $\frac{1}{4\varepsilon} - \frac{z}{\varepsilon} \leq 0$, so in this region we just need to check that $q(z)$ is not too negative. If $z \in [\frac{1}{4}, a^2]$, then $s(z) \geq 1$ and therefore $q(z) \geq 1$. Likewise, as we take d to be even, $q(z) \geq 1$ for $z \geq L^2$. For $z \in [a^2, L^2]$, we have $|q(z)| \leq 1$. Thus item (1) will be satisfied in this region so long as $a^2 \geq \frac{1}{4} + \varepsilon$. This holds as in the theorem statement $\varepsilon \leq 1/4$ and $a \geq 1/\sqrt{2}$.

With these easy cases taken care of, we turn to verify the first item for $z \in [0, \frac{1}{4}]$. To do this it suffices to choose d such that $q(1/4) \geq \frac{1}{4\varepsilon}$ as $q(z)$ is monotonically decreasing in the interval $[0, 1/4]$, since $T_d(y)$ is monotonically increasing for $y \geq 1$ by Fact 3.4 item (4). This condition is at odds with item (3). As the maximum of $q(z)$ in the interval $[1/4, L^2]$ is attained at $z = 1/4$, we can simultaneously satisfy item (3) by ensuring $q(1/4) \leq \frac{2}{\varepsilon}$. Thus we choose $d = d^*$ to be the least even number such that

$$q(1/4) = T_{d^*}\left(1 + \frac{2(a^2-1/4)}{L^2-a^2}\right) \geq \frac{1}{4\varepsilon}\,.$$

By this choice, item (1) is now satisfied. To verify item (3), we use Fact 3.4 item (3) to see the inequality $T_{s+2}(z) \leq 4z^2 T_s(z)$, valid for $z \geq 1$. Applying this we have

$$T_{d^*}\left(1 + \frac{2(a^2-1/4)}{L^2-a^2}\right) \leq \frac{1}{\varepsilon}\left(1 + \frac{2(a^2-1/4)}{L^2-a^2}\right)^2 \leq \frac{2}{\varepsilon}\,,$$

as $T_{d^*-2}\left(1 + \frac{2(a^2-1/4)}{L^2-a^2}\right) < \frac{1}{4\varepsilon}$ by definition, and $a \leq \sqrt{n}/8$.

Finally, we upper bound d^*. Let $\mu = 2(a^2 - 1/4)/(L^2 - a^2)$. We want $T_d(1+\mu) \geq \frac{1}{4\varepsilon}$. Using the fact that $T_d(1+\mu) \geq (1/2)(1+\sqrt{2\mu})^d$ for $\mu \geq 0$ by Fact 3.4 item (2), it suffices to take $d \geq \ln(\frac{1}{2\varepsilon})/\ln(1+\sqrt{2\mu})$.

As $\ln(1+y) \geq 2y/(2+y)$ for $y \geq 0$ and $\sqrt{2\mu} \leq 1$, it suffices to take $d \geq 3\ln(\frac{1}{2\varepsilon})/(2\sqrt{2\mu})$. Since $a \geq 1/\sqrt{2}$, and therefore $a^2 - 1/4 \geq a^2/2$, we have $\mu \geq a^2/L^2 = 4a^2/n^2$. Hence there is a d such that $T_d(1+\mu) \geq \frac{1}{4\varepsilon}$ satisfying

$$d \leq \left\lceil \frac{3n}{4\sqrt{2}\,a} \ln\left(\frac{1}{2\varepsilon}\right) \right\rceil .$$

We add 1 in the theorem statement for the additional requirement that the degree is even. ◀

Proof of Theorem 3.1. Fix $8/\sqrt{2n} \leq \delta \leq 1/4$, and let $\varepsilon = \delta/2$ and $a = \varepsilon\sqrt{n}/4$. Note that $1/\sqrt{2} \leq a \leq \sqrt{n}/8$ with these choices. Thus by Theorem 3.5, there is a polynomial p of degree at most $\lceil \frac{6\sqrt{n}}{\sqrt{2}\delta} \ln(\frac{1}{\delta}) \rceil + 1$ satisfying the three conditions of Theorem 3.5 with this value of a, ε.

Let $g(z) = (z - n/2)^2 - 1/4 + p(z - n/2)$ be a univariate polynomial, and consider the approximation to f given by $g(|x|)$. By construction g is globally nonnegative and thus (as it is univariate) is a sum of squares of polynomials of degree at most $\lceil \frac{3\sqrt{n}}{\sqrt{2}\delta} \ln(\frac{1}{\delta}) \rceil$. Let us examine the ℓ_1-error of the function $g(|x|)$ in approximating f. We divide the error into two cases: the error on strings whose Hamming weight is at most $n/2 - a$ or at least $n/2 + a$ (type I), and those whose Hamming weight is in the interval $[n/2 - a, n/2 + a]$ (type II).

As p is bounded by ε for $z \in [-n/2, -a] \cup [a, n/2]$ the ℓ_1-error over type I inputs is at most $\varepsilon \cdot 2^n$. The number of type II inputs is at most $(2a/\sqrt{n})2^n$, and the error on each is at most 2 as $p(z) \leq 2$ for $z \in [-n/2, n/2]$. Thus the total ℓ_1-error is $2^n \left(\varepsilon + \frac{4a}{\sqrt{n}}\right) = 2^n \cdot 2\varepsilon = \delta 2^n$. ◀

4 Proof complexity: Positivstellensatz refutations

Say that we have a system of polynomial equalities

$$f_1 = \cdots = f_m = 0,\ x_1^2 - x_1 = \cdots = x_n^2 - x_n = 0 \tag{4}$$

where each $f_i \in \mathbb{R}[x_1, \ldots, x_n]$. Because of the presence of the equalities $x_i^2 - x_i = 0$ (which force $x_i \in \{0, 1\}$), this is referred to as the *boolean* setting.

The Positivstellensatz [32] implies that the system (4) has no common solutions in $\mathbb{R}^n$ if and only if there are polynomials $g_1, \ldots, g_{m+n} \in \mathbb{R}[x_1, \ldots, x_n]$ and a sos polynomial h such that

$$\sum_{i=1}^{m} f_i g_i + \sum_{i=1}^{n} (x_i^2 - x_i) g_{m+i} = 1 + h\,. \tag{5}$$

Grigoriev and Vorobjov [14] define a proof system based on this principle.

▶ **Definition 4.1.** A *Positivstellensatz refutation* of the system (4) is given by a set of polynomials $\{g_1, \ldots, g_{m+n}, h\}$ which satisfy (5) and where h is a sum of squares. The *degree* of this refutation is

$$\max\{\deg(h), \max_{i \in [m]} \deg(f_i g_i), \max_{i \in [n]} \deg((x_i^2 - x_i) g_{m+i})\}\,.$$

By the Positivstellensatz, this proof system is sound and complete: a system is unsatisfiable if and only if it has a refutation of a certain degree. One may view the degree of a refutation as a measure of complexity.

4.1 Knapsack

The knapsack system is given by the equations

$$f = \sum_i x_i - r = 0,\ x_j^2 - x_j = 0 \text{ for } j = 1, \ldots, n\,. \tag{6}$$

If r is not an integer then this system has no solution: Grigoriev [12] shows the following theorem.

▶ **Theorem 4.2** (Grigoriev [12]). *Let $0 \le k \le (n-3)/2$ be an integer. If $k < r < n-k$, then any Positivstellensatz refutation of the system (6) has degree at least $2k+4$.*

We provide a simple proof of this in Appendix C using Blekherman's theorem.

Note that the equations for non-integer r correspond to a trivially easy (and obviously unsatisfiable) instance of the knapsack problem, where all items have weight 1. As mentioned in the introduction, this shows the weakness of the Positivstellensatz-based proof system: even to refute such easy instances it already needs polynomials of fairly high degree.

Grigoriev asked if this upper bound of $2k+4$ was tight. Later work of Grigoriev et al. [13] showed that the proof technique of [12] could not show a larger lower bound than $2k+4$. We show that there actually exist Positivstellensatz refutations of (6) of degree $2k+4$.

▶ **Theorem 4.3.** *Let $0 \le k \le n/2$ be an integer. For $k < r < k+1$, the system (6) has a Positivstellensatz refutation of degree $2k+4$.*

Let $\mathbf{x} = (x_1, \ldots, x_n)$ and $|\mathbf{x}| = \sum_{i=1}^n x_i$. A key role in the proof will be played by the polynomials

$$A_k(\mathbf{x}) = |\mathbf{x}|(|\mathbf{x}|-1)(|\mathbf{x}|-2)\cdots(|\mathbf{x}|-k+1)\,.$$

The function A_k can be computed with k queries by a natural extension of the Sampling Algorithm 1 and thus can be written as a sum-of-squares on the boolean cube of total degree $2k$. We go ahead and record this formally in the next lemma. Recall that the kth elementary symmetric polynomial is defined as

$$e_k(x_1, \ldots, x_n) = \sum_{i_1 < i_2 < \cdots < i_k} x_{i_1} x_{i_2} \cdots x_{i_k}\,.$$

▶ **Lemma 4.4.** *There exist polynomials $g_i(\mathbf{x})$ of degree at most $2k-2$ such that*

$$A_k(\mathbf{x}) = \sum_{i=1}^n (x_i^2 - x_i) g_i(\mathbf{x}) + (k!) e_k(x_1^2, \ldots, x_n^2)\,.$$

We give the proof of this in Appendix A.

Proof of Theorem 4.3. Rearranging Equation (5), we are looking for functions $g, g_1, \ldots, g_n$ of low degree and a low-degree sum-of-squares h such that

$$g(\mathbf{x})(|\mathbf{x}| - r) - 1 = h + \sum_i g_i(\mathbf{x})(x_i^2 - x_i)\,.$$

Notice that, for any g, the left-hand side will be negative when $|\mathbf{x}| = r$. By Lemma 4.4, A_{k+2} is of the form of the right-hand side. Since A_{k+2} has degree $2k+4$, and is also negative when $|\mathbf{x}| = r$, we try to find a polynomial $g(\mathbf{x})$ of degree at most $2k+3$ such that

$$g(\mathbf{x})\,(|\mathbf{x}| - r) - b = A_{k+2}(\mathbf{x})$$

for a positive constant b. Dividing g and A_{k+2} by b will then give us the required solution. Let $b = -r(r-1)\cdots(r-k)(r-k-1) > 0$. Then $|\mathbf{x}| - r$ divides $A_{k+2}(\mathbf{x}) + b$ and we can write $A_{k+2}(\mathbf{x}) + b = g(\mathbf{x})\,(|\mathbf{x}| - r)$ for some polynomial g of degree $2k+3$. ◀

5 Future work

We list a few questions for future work:

- Can we improve the lower bound of Theorem 1.1 for small ε? To match the upper bound for all k, it would suffice to show that $\text{sos-deg}_\varepsilon(f_1, \ell_\infty) = \Omega(\sqrt{n \log(1/\varepsilon)})$, which is very plausible by analogy with what is known for the n-bit OR function.
- Can we extend our results to all symmetric quadratic functions, or to even larger classes of symmetric functions?
- Can we find more applications of Blekherman's theorem (Theorem 2.3), in complexity theory, in quantum computing, or in optimization? Kurpisz et al. [17, Section 5] used their general reduction to univariate polynomials (already mentioned in Section 1.3.2), to show that strengthening the knapsack polytope with Wolsey's "Knapsack Covering Inequalities" and applying nearly $\log n$ rounds of the Lasserre hierarchy does not produce an SDP with integrality gap below $2 - o(1)$ (which is the integrality gap of the natural LP relaxation). Similar results may be obtainable using Blekherman's theorem.

Acknowledgments. We thank Mario Szegedy for early discussions on lower bounding the sos degree of f_1, Srinivasan Arunachalam for many useful discussions and comments, Greg Blekherman for discussions about [2], and Adam Kurpisz for alerting us to [17]. We thank the authors of the software SDPT3 [33] and YALMIP [20], which we used to compute the error of sos polynomials in approximating f_k on small examples. TL would like to thank Savaroo Ranch for their hospitality and BBQ during the course of this research.

References

1 R. Beals, H. Buhrman, R. Cleve, M. Mosca, and R. de Wolf. Quantum lower bounds by polynomials. *Journal of the ACM*, 48(4):778–797, 2001. Earlier version in FOCS'98.

2 G. Blekherman. Symmetric sums of squares on the hypercube. *Manuscript in preparation*, 2015.

3 G. Blekherman and C. Riener. Symmetric nonnegative forms and sums of squares, 2012. arXiv:1205.3102.

4 G. Brassard, P. Høyer, M. Mosca, and A. Tapp. Quantum amplitude amplification and estimation. In *Quantum Computation and Quantum Information: A Millennium Volume*, volume 305 of *AMS Contemporary Mathematics Series*, pages 53–74. AMS, 2002. quant-ph/0005055.

5 H. Buhrman, R. Cleve, R. de Wolf, and Ch. Zalka. Bounds for small-error and zero-error quantum algorithms. In *Proceedings of 40th IEEE FOCS*, pages 358–368, 1999. cs.CC/9904019.

6 T. Ceccherini-Silberstein, F. Scarabotti, and F. Tolli. *Harmonic analysis on finite groups: representation theory, Gelfand pairs and Markov chains*, volume 108. Cambridge University Press, 2008.

7 A. Childs and T. Lee. Optimal quantum adversary lower bounds for ordered search. In *Proceedings of 35th International Colloquium on Automata, Languages and Programming (ICALP'08)*, pages 869–880, 2008.

8 Y. Filmus and E. Mossel. Harmonicity and invariance on slices of the boolean cube. In *Proceedings of 31st CCC*, 2016. arXiv:1507.02713.

9 S. Fiorini, S. Massar, S. Pokutta, H. R. Tiwary, and R. de Wolf. Exponential lower bounds for polytopes in combinatorial optimization. *Journal of the ACM*, 16(2), 2015. arxiv/1111.0837. Earlier version (under a different name) in STOC'12.

10 C. Godsil. Association schemes, 2010. Lecture notes available on `http://www.math.uwaterloo.ca/~cgodsil/pdfs/assoc2.pdf`, version 1.

11 M. de Graaf and R. de Wolf. On quantum versions of the Yao principle. In *Proceedings of 19th Annual Symposium on Theoretical Aspects of Computer Science (STACS'02)*, pages 347–358, 2002. quant-ph/0109070.

12 D. Grigoriev. Complexity of Positivstellensatz proofs for the knapsack. *Computational Complexity*, 10(2):139–154, 2001.

13 D. Grigoriev, E. Hirsch, and D. Pasechnik. Complexity of semialgebraic proofs. *Moscow Mathematical Journal*, 2(4):647–679, 2002.

14 D. Grigoriev and N. Vorobjov. Complexity of Null- and Positivstellensatz proofs. *Annals of Pure and Applied Logic*, 113:153–160, 2001.

15 L. K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of 28th ACM STOC*, pages 212–219, 1996. quant-ph/9605043.

16 J. Kaniewski, T. Lee, and R. de Wolf. Query complexity in expectation. In *Proceedings of 42nd International Colloquium on Automata, Languages and Programming (ICALP'15)*, pages 761–772, 2015. arXiv:1411.7280.

17 A. Kurpisz, S. Leppänen, and M. Mastrolilli. Sum-of-squares hierarchy lower bounds for symmetric formulations. In *Proceedings of 18th IPCO*, 2016. arXiv:1407.1746.

18 J. B. Lasserre. Global optimization with polynomials and the problem of moments. *SIAM Journal on Optimization*, 11(3):796–817, 2001.

19 J. R. Lee, P. Raghavendra, and D. Steurer. Lower bounds on the size of semidefinite programming relaxations. In *Proceedings of 47th ACM STOC*, pages 567–576, 2015.

20 J. Löfberg. YALMIP: A toolbox for modeling and optimization in Matlab. In *Proceedings of the CACSD conference*, 2004.

21 M. Minsky and S. Papert. *Perceptrons*. MIT press, Cambridge, MA, 1968. Second, expanded edition 1988.

22 N. Nisan and M. Szegedy. On the degree of Boolean functions as real polynomials. *Computational Complexity*, 4:301–313, 1994.

23 R. O'Donnell and R. Servedio. New degree bounds for polynomial threshold functions. *Combinatorica*, 30(3):327–358, 2010.

24 P. Parrilo. *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. PhD thesis, California Institute of Technology, 2000.

25 R. Paturi. On the degree of polynomials that approximate symmetric boolean functions. In *Proceedings of 24th ACM STOC*, pages 468–474, 1992.

26 M. Petkovšek, H. Wilf, and D. Zeilberger. *"A=B"*. Springer Science & Business Media, 1997.

27 T. Rivlin. *An introduction to the approximation of functions*. Dover publications, inc., 1969.

28 A. Sherstov. Approximate inclusion-exclusion for arbitrary symmetric functions. *Computational Complexity*, 18:219–247, 2009.

29 A. Sherstov. The pattern matrix method. *SIAM Journal on Computing*, 40(6):1969–2000, 2011.

30 Y. Shi and Y. Zhu. Quantum communication complexity of block-composed functions. *Quantum information and computation*, 9(5,6):444–460, 2009. URL: `http://www.rintonpress.com/xxqic9/qic-9-56/0444-0460.pdf`.

31 N. Z. Shor. An approach to obtaining global extremums in polynomial mathematical programming problems. *Cybernetics*, 23:695–700, 1987.

32 G. Stengle. A Nullstellensatz and a Positivstellensatz in semialgebraic geometry. *Mathematische Annalen*, 207:87–97, 1974.

33 K.C. Toh, M.J. Todd, and R.H. Tutuncu. SDPT3 – a Matlab software package for semidefinite programming. *Optimization methods and software*, 11:545–581, 1999.
34 R. de Wolf. A note on quantum algorithms and the minimal degree of epsilon-error polynomials for symmetric functions. *Quantum Information and Computation*, 8:943–950, 2008.

A Proof of Lemma 4.4

Recall that

$$A_k(x_1,\ldots,x_n) = |\mathbf{x}|(|\mathbf{x}|-1)\cdots(|\mathbf{x}|-k+1)\,.$$

We first prove two claims.

▶ **Claim A.1.** *There exist polynomials $g_i(\mathbf{x})$ of degree at most $k-1$ such that*

$$A_k(x_1,\ldots,x_n) = \sum_{i=1}^{n}(x_i^2-x_i)g_i(\mathbf{x}) + (k!)e_k(\mathbf{x})\,.$$

Proof. We prove the claim by induction on k. When $k=1$ then $A_1 = e_1$, and the claim follows by setting all g_i to be 0.

Now suppose the claim is true up to k. Then, using the induction hypothesis to rewrite A_k,

$$\begin{aligned} A_{k+1}(\mathbf{x}) = A_k(\mathbf{x})\cdot(e_1(\mathbf{x})-k) &= \left(\sum_{i=1}^{n}(x_i^2-x_i)g_i(\mathbf{x}) + k!e_k(\mathbf{x})\right)(e_1(\mathbf{x})-k) \\ &= \sum_{i=1}^{n}(x_i^2-x_i)h_i(\mathbf{x}) + k!e_k(\mathbf{x})(e_1(\mathbf{x})-k)\,, \end{aligned}$$

where each $h_i(\mathbf{x}) = g_i(\mathbf{x})\cdot(e_1(\mathbf{x})-k)$ is of degree at most k. We now focus on

$$e_k(\mathbf{x})(e_1(\mathbf{x})-k) = \sum_{\substack{S\subseteq[n]\\|S|=k}}\prod_{i\in S}x_i\cdot(e_1(\mathbf{x})-k)\,.$$

A term in this sum corresponding to the subset S can be rewritten as

$$\prod_{i\in S}x_i\left(\sum_{i\in S}x_i + \sum_{i\notin S}x_i - k\right) = \sum_{i\in S}(x_i^2-x_i)\prod_{j\in S, j\neq i}x_j + \sum_{i\notin S}x_i\prod_{j\in S}x_j$$

Summing over all terms of this form gives $(k+1)!e_{k+1}(\mathbf{x}) + \sum_i(x_i^2-x_i)\cdot f_i(\mathbf{x})$, where $f_i(\mathbf{x})$ is of degree at most $k-1$, proving the claim. ◀

To complete the proof, we now need to show that $e_d(\mathbf{x})$ is a sum of squares of total degree $2d$ modulo the ideal $\langle x_1^2-x_1,\ldots,x_n^2-x_n\rangle$. To do this, it suffices to show the same for $\prod_{i=1}^{d}x_i$, which we do in the next claim.

▶ **Claim A.2.** *Fix a natural number d. Then there are polynomials $g_i\in\mathbb{R}[x_1,\ldots,x_d]$ for $i=1,\ldots,d$ such that*

$$\prod_{i=1}^{d}x_i^2 - \prod_{i=1}^{d}x_i = \sum_{i=1}^{d}(x_i^2-x_i)g_i(x_1,\ldots,x_d)\,,$$

and each g_i is of degree at most $2d-2$.

Proof. We write $x_1^2x_2^2\cdots x_d^2 - x_1x_2\cdots x_d$ as a telescoping sum. We use the convention that the product over the empty set is 1.

$$\prod_{i=1}^{d} x_i^2 - \prod_{i=1}^{d} x_i = \sum_{j=1}^{d}\left(\prod_{i<j} x_i \prod_{i\geq j} x_i^2 - \prod_{i\leq j} x_i \prod_{i>j} x_i^2\right)$$
$$= \sum_{j=1}^{d}(x_j^2 - x_j)\prod_{i<j} x_i \prod_{i>j} x_i^2$$

This is of the desired form, and it can be seen that each multiplier of $x_j^2 - x_j$ is of degree at most $2d-2$. ◀

We put these claims together to prove Lemma 4.4, which we restate here.

▶ **Lemma A.3.** *There exist polynomials $g_i(\mathbf{x})$ of degree at most $2k-2$ such that*

$$A_k(\mathbf{x}) = \sum_{i=1}^{n}(x_i^2 - x_i)g_i(\mathbf{x}) + (k!)e_k(x_1^2,\ldots,x_n^2)\,.$$

Proof. By Claim A.1 we can write $A_k(x_1,\ldots,x_n) = \sum_{i=1}^{n}(x_i^2 - x_i)g_i(\mathbf{x}) + (k!)e_k(\mathbf{x})$ where each $g_i(\mathbf{x})$ is of degree at most $k-1$. Now by Claim A.2

$$e_k(\mathbf{x}) = e_k(x_1^2,\ldots,x_n^2) + \sum_{i=1}^{n}(x_i^2 - x_i)\cdot f_i(\mathbf{x})\,,$$

where each $f_i(\mathbf{x})$ is of degree at most $2k-2$. This proves the lemma. ◀

B Blekherman's theorem

Blekherman and Riener [3] made a general study of the relationship between symmetric nonnegative forms and symmetric sums of squares. Subsequently, Blekherman [2] considered the special case of polynomials that are nonnegative on the hypercube, and gave a very useful decomposition of such polynomials. We include a proof here of a special case of his theorem.

The technique used for our proof is a novel decomposition of functions on the hypercube using the kernels of certain differential operators. A similar decomposition was independently discovered by Filmus and Mossel [8] who use it to prove an invariance principle for low-degree functions on slices of the boolean hypercube.

Let $[n]$ denote the set of integers $\{1,2,\ldots,n\}$. The ideal $\mathcal{I} := \langle x_i^2 - x_i : i \in [n]\rangle$ consists of polynomials that are identically zero on the hypercube $\mathcal{H} = \{0,1\}^n$. Let $L_t = \mathbb{R}[x]_t/\mathcal{I}$ be the space of degree-t homogeneous multilinear polynomials on n variables. The $\binom{n}{t}$ monomials x^S where $S \subseteq [n], |S| = t$, form a basis for L_t. The correspondence between set S and monomial x^S can be used to map degree-t polynomials to linear combinations of t-subsets of $[n]$. Here polynomial $p(x)$ corresponds to the $\binom{n}{t}$-dimensional vector of the coefficients of its monomials, say in lexicographic order.

Let $M_t = \mathbb{R}[x]_{\leq t}/\mathcal{I}$ denote the space of n-variate polynomials of degree at most t on the hypercube. Given $x \in \mathbb{R}^n$, the sum $\sum_{i\in[n]} x_i$ is denoted by $|x|$.

B.1 Kernels of the operators W_t

For $t \geq 0$, define the linear operator W_t that acts by summing over the partial derivatives of a degree-t polynomial,

$$W_t p(x) = \left(\sum_{i \in [n]} \frac{\partial}{\partial x_i} \right) p(x) \,. \tag{7}$$

For $t \geq 1$ the operator $W_t : L_t \to L_{t-1}$ is represented by a matrix with rows and columns indexed by $S, T \subset [n]$ with $|S| = t-1$ and $|T| = t$ respectively, and entry $(W_t)_{S,T} = 1$ if $S \subset T$ and 0 otherwise. The adjoint operator $W_t^T : L_{t-1} \to L_t$ acts as multiplication by $|x| - t + 1$ on each degree-$(t-1)$ monomial (and by linear extension on all of L_{t-1}),

$$W_t^T x^S = \sum_{i \notin S} x^{S \cup \{i\}} = x^S (|x| - t + 1) \,. \tag{8}$$

Note that we used the hypercube constraints $x_i^2 = x_i$ to derive the second equality in Equation (8). Our goal in this section is to bound the dimension of $\mathrm{Ker}(W_t)$ and find an explicit basis for these spaces.

We relate $\mathrm{Ker}(W_t)$ to the eigenspaces of the Johnson graphs. The Johnson graph $J(n,t)$ has $\binom{n}{t}$ vertices corresponding to the t-subsets $S \subset [n], |S| = t$, with subsets S, T connected by an edge if and only if $|S \cap T| = t - 1$. The adjacency matrix of $J(n,t)$ is denoted by $A_J(n,t)$. The following lemma computes the spectrum of $A_J(n,t)$; it can be found in Godsil's notes [10], but we include a proof here as these notes are no longer online.

▶ **Theorem B.1.** *The eigenvalues of $A_J(n,t)$ are $t(n-t) - i(n+1-i)$ with multiplicity $\binom{n}{i} - \binom{n}{i-1}$ for $i = \{0, 1, \ldots, t\}$ and $t \leq (n+1)/2$.*

Proof. We proceed by induction on n and t. For the base case, note that the Johnson graph $J(n,1)$ is the complete graph on n vertices. The corresponding adjacency matrix $A_J(n,1)$ has eigenvalue -1 with multiplicity $(n-1)$ and $(n-1)$ with multiplicity 1, thus the theorem is true for $n = 1$.

We obtain the spectrum of $A_J(n,t)$ in terms of the spectrum of $A_J(n,t-1)$. Computing the entries of $W_t^T W_t$ and $W_t W_t^T$ it follows that,

$$\begin{aligned} W_t^T W_t &= tI + A_J(n,t) \\ W_t W_t^T &= (n-t+1)I + A_J(n,t-1) \,. \end{aligned} \tag{9}$$

The non-zero eigenspaces of $W_t^T W_t$ correspond to those of $W_t W_t^T$, so if v is an eigenvector for $A_J(n,t-1)$ with eigenvalue λ_i then $W_t^T v$ is an eigenvector for $A_J(n,t)$ with eigenvalue $\lambda_i + n - 2t + 1$. By the induction hypothesis, $(t-1)(n-t+1) - i(n+1-i)$ is an eigenvalue for $A_J(n,t-1)$ with multiplicity $\binom{n}{i} - \binom{n}{i-1}$ for $i \in [t-1]$. Adding $n - 2t + 1$, it follows that $t(n-t) - i(n+1-i)$ is an eigenvalue for $A_J(n,t)$ with the same multiplicity.

The induction hypothesis also implies that $W_t W_t^T = (n-t+1)I + A_J(n,t-1)$ has rank $\binom{n}{t-1}$ as it is positive semidefinite and the smallest eigenvalue is $n - 2t + 2 > 0$. Hence the $\binom{n}{t}$-dimensional matrix $W_t^T W_t$ has rank $\binom{n}{t-1}$, so its kernel has dimension $\binom{n}{t} - \binom{n}{t-1}$. This implies that $A_J(n,t)$ has an eigenspace of dimension $\binom{n}{t} - \binom{n}{t-1}$ with eigenvalue $-t = t(n-t) - t(n+1-t)$. ◀

The following corollary computes the dimension of $\mathrm{Ker}(W_t)$.

▶ **Lemma B.2.** *$Dim(Ker(W_t)) = \binom{n}{t} - \binom{n}{t-1}$ for $t \leq (n+1)/2$.*

Proof. $\mathrm{Dim}(\mathrm{Ker}(W_t)) = \binom{n}{t} - \mathrm{rank}(W_t^T)$ by definition, and from the above proof it follows that $\mathrm{rank}(W_t^T) = \mathrm{rank}(W_t W_t^T) = \binom{n}{t-1}$ for $t \leq (n+1)/2$. ◀

We next compute an explicit basis for $\mathrm{Ker}(W_t)$, viewed as a subspace of L_t. We recall the notion of a standard Young tableau of shape $(n-t, t)$ to describe the basis.

▶ **Definition B.3.** A standard Young tableau $\mathcal{U}$ of shape $(n-t, t)$ is an arrangement of $[n]$ in an array with two rows of size $n-t$ and t respectively, such that each row and column is sorted in ascending order.

The basis for $\mathrm{Ker}(W_t)$ described by the following theorem will be used for computations in the following sections. Note that polynomials $p_{\mathcal{U}}$ in this basis evaluate to 0 for all $x \in \{0,1\}^n$ with $|x| \in \{0, 1, \ldots, t-1\} \cup \{n, n-1, \ldots, n-t+1\}$.

▶ **Theorem B.4.** *For $t \leq n/2$ and $\mathcal{A} = (a(1), a(2), \ldots, a(2t))$ an array of distinct elements $a(i) \in [n]$, define the polynomial $p_{\mathcal{A}}(x) := \prod_{i \in [t]} (x_{a(2i-1)} - x_{a(2i)})$.*

The polynomials $p_{\mathcal{U}}(x)$, where $(u(2i-i), u(2i))$ for $i \in [t]$ are the entries of the i-th column of a standard $(n-t, t)$ Young tableau $\mathcal{U}$, form a basis for Ker(W_t).

Proof. We first show that for all $|\mathcal{A}| = 2t$, the degree-t polynomial $p_{\mathcal{A}}(x)$ belongs to the kernel of W_t. Computing the partial derivatives of $p_{\mathcal{A}}(x)$,

$$\frac{\partial}{\partial x_j} p_{\mathcal{A}}(x) = \begin{cases} p_{\mathcal{A}}(x)/(x_{a(2i-1)} - x_{a(2i)}) \text{ if } j = a(2i-1) \\ -p_{\mathcal{A}}(x)/(x_{a(2i-1)} - x_{a(2i)}) \text{ if } j = a(2i) \\ 0 \text{ otherwise} \end{cases} \tag{10}$$

Summing over the partial derivatives and using Equation (7) it follows that $W_t p_{\mathcal{A}}(x) = 0$.

The set of polynomials $\{p_{\mathcal{A}}(x) : |\mathcal{A}| = 2t\}$ is not linearly independent. The straightening algorithm for Young tableaux (see for example Section 10.5 in [6]) shows that the polynomials $p_{\mathcal{U}}(x)$ where $(u(2i-i), u(2i))$ for $i \in [t]$ are entries of the i-th column of a standard $(n-t, t)$ Young tableau $\mathcal{U}$ form a basis for $\mathrm{Span}\{p_{\mathcal{A}}(x) : |\mathcal{A}| = 2t\}$. A simple counting argument or the hook length formula [6] shows that the number of such $\mathcal{U}$ is $\binom{n}{t} - \binom{n}{t-1}$. These $p_{\mathcal{U}}$ together thus span a space of dimension $\binom{n}{t} - \binom{n}{t-1}$, which is $\mathrm{Dim}(\mathrm{Ker}(W_t))$ by Lemma B.2. Hence the $p_{\mathcal{U}}$ form a basis for $\mathrm{Ker}(W_t)$. ◀

B.2 Polynomial decompositions

The action of the operators W_t yields the decomposition $L_t = \mathrm{Ker}(W_t) \oplus \mathrm{Im}(W_t^T)$. Applying this decomposition iteratively we obtain the following theorem,

▶ **Theorem B.5.** *A polynomial $p(x) \in L_t$ can be decomposed as*

$$\begin{aligned} p(x) &= p_t(x) + (|x| - t + 1) p_{t-1}(x) + \cdots + (|x| - t + 1) \cdots (|x| - 1)|x| p_0(x) \\ &= p_t(x) + \sum_{i=1}^{t} p_{t-i}(x) \prod_{j=1}^{i} (|x| - t + j) \end{aligned} \tag{11}$$

where $p_{t-i}(x) \in$ Ker(W_{t-i}).

Proof. We proceed by induction on t. For the base case $t = 0$, observe that a degree-0 polynomial belongs to $\mathrm{Ker}(W_0)$. For the inductive step, a polynomial $p(x) \in L_t$ can be

written as $p_t(x) + q(x)$ where $p_t(x) \in \mathrm{Ker}(W_t)$ and $q(x) \in \mathrm{Im}(W_t^T)$. The action of W_t^T on polynomials in L_{t-1} is described by Equation (8): for all $g(x) \in L_{t-1}$ we have

$$W_t^T g(x) = (|x| - t + 1)g(x)\,. \tag{12}$$

As $q(x) \in \mathrm{Im}(W_t^T)$, it can be factored as $q(x) = (|x| - t + 1)h(x)$ where $h(x) \in L_{t-1}$. The result follows using the induction hypothesis for $g(x)$. ◀

Applying the above theorem to the subspaces L_j ($j \in \{0, \ldots, t\}$) that are contained in M_t and collecting the terms corresponding to $\mathrm{Ker}(W_j)$, we obtain the following decomposition for polynomials in M_t.

▶ **Corollary B.6.** *A polynomial $p(x) \in M_t$ can be decomposed as $p(x) = \sum_{j=0}^{t} q_j(x)$, where*

$$q_j(x) = \sum_{0 \le i \le t-j} |x|^i p_{ij}(x) \tag{13}$$

such that each $p_{ij}(x) \in Ker(W_j)$.

Proof. A polynomial $p(x) \in M_t$ can be written as $p(x) = \sum_{i=0}^{t} p_i(x)$ where $p_i(x) \in L_i$ is the homogeneous degree-i component of p. Applying Theorem B.5 to each $p_i(x)$ and collecting all the terms over the Equations (11) with prefactors in $\mathrm{Ker}(W_j)$, we obtain a decomposition $p(x) = \sum_{j \in [t]} q'_j(x)$ such that

$$\begin{aligned} q'_j(x) = p'_{jj}(x) + p'_{j+1,j}(x)(|x| - j) + p'_{j+2,j}(x)(|x| - j)(|x| - j + 1) + \cdots \\ \cdots + p'_{t,j}(x)(|x| - j)(|x| - j + 1)\cdots(|x| - t + 1)\,. \end{aligned} \tag{14}$$

Note that the indices in the above equation increase, because $\mathrm{Ker}(W_j)$ occurs in the decompositions of $p_i(x)$ for $i \ge j$. Let $p_{ij}(x)$ be the coefficient of $|x|^i$ for $0 \le i \le t - j$ in the above expression. This $p_{ij}(x)$ is a linear combination of polynomials $p'_{ij}(x) \in \mathrm{Ker}(W_j)$ and therefore also lies in $\mathrm{Ker}(W_j)$. The decomposition in Equation (13) follows. ◀

B.3 Symmetrization and Blekherman's theorem

The symmetric group S_n acts on the polynomial ring M_n by permuting the indices of the monomials. The subspace of symmetric polynomials in M_n that are invariant under the action of S_n is denoted by Λ_n. The operator $\mathrm{Sym} : M_n \to \Lambda_n$ maps a polynomial to its symmetrization,

$$\mathrm{Sym}(p)(x) := \frac{1}{n!} \sum_{\sigma \in S_n} p(\sigma x)\,. \tag{15}$$

The symmetrization of degree k monomials evaluates to a univariate polynomial in $|x|$ over M_n.

▶ **Lemma B.7.** *Let $m_k(x) = x_1 x_2 \cdots x_k$ be a degree-k monomial, then the following identity is true in the ring M_n,*

$$Sym(m_k)(x) = \frac{|x|(|x| - 1)\cdots(|x| - k + 1)}{n(n-1)\cdots(n-k+1)}\,. \tag{16}$$

Proof. We proceed by induction on k, for $k = 1$ the result is clearly true. Let x^S be an arbitrary degree k monomial. There are $k!(n-k)!$ permutations $\sigma \in S_n$ such that $\sigma(x_1 x_2 \cdots x_k) = x^S$, thus $\mathrm{Sym}(m_k)(x)$ evaluates to

$$\mathrm{Sym}(m_k)(x) = \frac{1}{\binom{n}{k}} \sum_{|S|=k} x^S \,. \tag{17}$$

In order to to express $\mathrm{Sym}(m_k)(x)$ in terms of $\mathrm{Sym}(m_{k-1})(x)$, we write the above equation in terms of the operator W_k^t from Section B.1,

$$\mathrm{Sym}(m_k)(x) = \frac{1}{\binom{n}{k}} W_k^T \left(\frac{1}{k} \sum_{|U|=k-1} x^U \right) = \frac{(|x|-k+1)}{(n-k+1)} Sym(m_{k-1})(x) \,. \tag{18}$$

The second equality follows from Equation (8) and the expression for $\mathrm{Sym}(m_{k-1})$ in Equation (17). The result follows from the induction hypothesis. ◀

The above lemma shows that $\mathrm{Sym}(p)$ for polynomials $p \in M_n$ can be viewed as a univariate polynomial in $|x|$ by extending the mapping given by Lemma B.7 to all $p \in M_n$. We denote the univariate polynomial thus obtained by $\mathrm{Sym}^{uni}(p)$ to disambiguate from the multivariate polynomial in Equation (15).

We can define an inner product on polynomials $p, q \in L_t$ by treating them as vectors of coefficients: if $p(x) = \sum_{|S|=t} p_S x^S$ and $q(x) = \sum_{|S|=t} q_S x^S$ then

$$\langle p|q \rangle := \sum_{S \subseteq [n], |S|=t} p_S q_S \,. \tag{19}$$

The symmetrization of the product of polynomials in $\mathrm{Ker}(W_t)$ can be expressed in terms of this inner product.

▶ **Lemma B.8.** *If $p, q \in Ker(W_t)$ for some $t \leq n/2$, then:*

$$Sym(pq)(x) = \langle p|q \rangle \frac{(n-2t)!}{n!} \prod_{0 \leq i < t} (|x|-i)(n-|x|-i) \,. \tag{20}$$

Proof. Theorem B.4 shows that $\mathrm{Ker}(W_t)$ has a basis consisting of polynomials $p_{\mathcal{U}}(x)$ such that $p_{\mathcal{U}}(x) = 0$ for all $x \in \{0,1\}^n$ with $|x| \in \{0, 1, \ldots, t-1\} \cup \{n, n-1, \ldots, n-t+1\}$. Consider such an x. Evaluating $\mathrm{Sym}(pq)$ at x using Equation (15) by expanding p and q in the basis given by Theorem B.4, it follows that $\mathrm{Sym}^{uni}(pq)(\alpha) = 0$ for all $\alpha \in \{0, 1, \ldots, t-1\} \cup \{n, n-1, \ldots, n-t+1\}$. Lemma B.7 shows that $\mathrm{Sym}(pq)(x)$ is a univariate polynomial Sym^{uni} in $|x|$ of degree at most $2t$, hence

$$\mathrm{Sym}(pq)(x) = \lambda \prod_{0 \leq i < t} (|x|-i)(n-|x|-i) \,. \tag{21}$$

for some $\lambda \in \mathbb{R}$. Below we determine λ by evaluating $\mathrm{Sym}(pq)$ for $x \in \{0,1\}^n$ such that $|x| = t$.

We compute $\mathrm{Sym}(pq)(x)$ by evaluating the sum $\sum_{\sigma \in S_n} p(\sigma x) q(\sigma x)$ in Equation (15). As p, q are homogeneous degree-t polynomials, for each x with $|x| = t$ there is a unique $S \subset [n]$, $|S| = t$, such that $p(x) = p_S$ and $q(x) = q_S$. In other words, x sets exactly one degree-t monomial x^S to 1 and all others to 0. There are $t!(n-t)!$ different $\sigma \in S_n$ such that $\sigma(x)$ sets the same monomial to 1. The symmetrization $\mathrm{Sym}(pq)(x)$ therefore evaluates to

$$\mathrm{Sym}(pq)(x) = \frac{1}{n!} \sum_{\sigma \in S_n} p(\sigma x) q(\sigma x) = \frac{t!(n-t)!}{n!} \sum_{|S|=t} p_S q_S \,. \tag{22}$$

$\mathrm{Sym}(pq)(x)$ also evaluates to $\lambda \prod_{0 \le i < t}(t-i)(n-t-i)$. Equating the two expressions we have:

$$\lambda t! \prod_{0 \le i < t} (n-t-i) = \frac{\langle p|q\rangle t!(n-t)!}{n!} \tag{23}$$

which implies $\lambda = \frac{\langle p|q\rangle (n-2t)!}{n!}$, and the theorem follows. ◀

We next show that the symmetrization of the product of polynomials $p \in \mathrm{Ker}(W_t), q \in \mathrm{Ker}(W_{t'})$ evaluates to 0 if $t \neq t'$. The following lemma is used for the proof in Lemma B.10.

▶ **Lemma B.9.** *If $p(x) = \prod_{i\in[k]}(x_i - x_{i+1})q(x)$ for some odd k, and $q(x)$ is a polynomial that does not depend on variables $x_1, \ldots, x_{k+1}$, then Sym$(p) = 0$.*

Proof. It suffices to show that $\mathrm{Sym}(p)(x) = 0$ for all $x \in \{0,1\}^n$, because a multilinear polynomial that is 0 on the hypercube is identically equal to 0. Define the involution $\sigma \to \overline{\sigma}$ on S_n by setting $\overline{\sigma}(i) = \sigma(i+1)$ if $i \in [k+1]$ is odd, $\overline{\sigma}(i) = \sigma(i-1)$ if $i \in [k+1]$ is even, and $\overline{\sigma}(i) = \sigma(i)$ for $i > k+1$. It follows that $\overline{\sigma}$ is an involution as $k+1$ is even and it acts by swapping the pairs $(\sigma(2i-1), \sigma(2i))$ for $i \in [(k+1)/2]$. This involution partitions S_n into pairs $(\sigma, \overline{\sigma})$, and hence

$$\mathrm{Sym}(p)(x) = \frac{1}{n!} \sum_{(\sigma,\overline{\sigma})} (p(\sigma x) + p(\overline{\sigma} x)) = 0. \tag{24}$$

The second equality follows as $p(\overline{\sigma}x) = -p(\sigma x)$ for all $x \in \{0,1\}^n$ and $\sigma \in S_n$. ◀

▶ **Lemma B.10.** *If $p \in Ker(W_t)$ and $q \in Ker(W_{t'})$ for $n/2 \ge t > t'$, then Sym$(pq) = 0$.*

Proof. It suffices to prove the statement for polynomials $p = p_{\mathcal{U}}$ and $q = q_{\mathcal{V}}$ belonging to the bases for $\mathrm{Ker}(W_t)$ and $\mathrm{Ker}(W_{t'})$ constructed in Theorem B.4. The arrays $\mathcal{U}, \mathcal{V}$ define matchings $M(\mathcal{U}) = \bigcup_{i\in[t]}(u(2i-1), u(2i))$ and $M(\mathcal{V}) = \bigcup_{i\in[t']}(v(2i-1), v(2i))$ on $[n]$ of size t and t' respectively. The product $p_{\mathcal{U}} q_{\mathcal{V}} = \prod_{(a,b)\in M(\mathcal{U})\cup M(\mathcal{V})}(x_a - x_b)$. If $M(\mathcal{U}) \cup M(\mathcal{V})$ contains an odd-length path as an induced subgraph, then we can use Lemma B.9 to conclude that $\mathrm{Sym}(p_{\mathcal{U}} q_{\mathcal{V}}) = 0$.

It suffices to show that the union of two matchings of different sizes contains an odd-length path as an induced subgraph. The connected components of a union of two distinct matchings on $[n]$ either form even-length cycles or paths. Color the edges in $M(\mathcal{U})$ red and the edges in $M(\mathcal{V})$ blue. The number of red edges t is greater than blue edges t', so there must be at least one connected component that is an odd-length path, as even-length paths and cycles have an equal number of red and blue edges. ◀

The preceding lemmas allow us to give a proof of Blekherman's result [2] on the symmetrization of sum-of-squares polynomials on the hypercube.

▶ **Theorem B.11** (Blekherman)**.** *The symmetrization of the square of polynomial $p \in M_t$ for $t \le n/2$ can be decomposed as*

$$Sym(p^2)(x) = \sum_{j=0}^{t} p_{t-j}(|x|) \left(\prod_{0 \le i < j} (|x| - i)(n - |x| - i) \right) \tag{25}$$

where p_{t-j} is a univariate polynomial that is the sum of squares of polynomials of degree at most $t-j$.

Proof. Consider the representation of the polynomial $p(x) = \sum_{j=0}^{t} q_j(x)$ given by Corollary B.6,

$$q_j(x) = \sum_{0 \le k \le t-j} |x|^k p_{kj}(x) \tag{26}$$

where the polynomials $p_{kj}(x) \in \mathrm{Ker}(W_j)$. Lemma B.10 shows that $\mathrm{Sym}(p_{kj}p_{k'j'}) = 0$ if $j \neq j'$, hence $\mathrm{Sym}(p^2)$ can be decomposed as

$$\mathrm{Sym}(p^2) = \sum_{j=0}^{t} \mathrm{Sym}\left(q_j^2\right) . \tag{27}$$

Expanding the term $\mathrm{Sym}\left(q_j^2\right)$ using Lemma B.8, we have

$$\begin{aligned} \sum_{0 \le k,l \le t-j} \mathrm{Sym}(|x|^{k+l} p_{kj} p_{lj}) &= c \left(\prod_{0 \le i < j} (|x| - i)(n - |x| - i) \right) \sum_{0 \le k,l \le t-j} \langle p_{kj} | p_{lj} \rangle |x|^{k+l} \\ &= c \left(\prod_{0 \le i < j} (|x| - i)(n - |x| - i) \right) \mathbf{x}^T P \mathbf{x} \end{aligned} \tag{28}$$

where c is a constant independent of $|x|$, $P \in \mathbb{R}^{(t-j+1)\times(t-j+1)}$ is the matrix with entries $P_{kl} = \langle p_{kj} | p_{lj} \rangle$, and $\mathbf{x} \in \mathbb{R}^{t-j+1}$ is the vector with entries $(1, |x|, |x|^2, \ldots, |x|^{t-j})$. The matrix P is positive semidefinite, hence the polynomial $p_{t-j}(|x|)$ corresponding to the quadratic form $\mathbf{x}^t P \mathbf{x}$ is a sum of squares of polynomials in $|x|$ of degree at most $t - j$. The theorem follows. ◀

Note that the proof is constructive, as it provides a way to compute the terms in the decomposition by projecting onto the eigenspaces W_t of the Johnson scheme. For example, the first term $p_t(|x|)$ in (25) is in fact $\mathrm{Sym}^{uni}(p)^2$ as the Sym operator maps $\mathrm{Ker}(W_j)$ to 0 for all $j > 0$.

▶ **Corollary B.12.** *The polynomial $p_t(|x|)$ in Theorem B.11 is $Sym^{uni}(p)^2$.*

A symmetric function f that is the sum of squares of polynomials of degree $d \le n/2$ is a sum of terms $\mathrm{Sym}(p^2)$ for n-variate polynomials p of degree $d \le n/2$. Applying Theorem B.11 for $t = d$ we obtain Blekherman's result as stated in Theorem 2.3.

Note that Theorem B.11 applies to the setting where $deg(p(x)) \le n/2$, this suffices for our applications. Blekherman's theorem in [2] is valid for all degrees modulo the ideal $I = \langle \prod_{0 \le i \le n} (|x| - i) \rangle$.

C Grigoriev's knapsack lower bound

We now see how Blekherman's theorem can be easily used to reprove Grigoriev's lower bound on the degree of Positivstellensatz refutations of knapsack (Theorem 4.2). A Positivstellensatz refutation of the knapsack system of equations (1) with parameter r consists of polynomials $g, g_1, \ldots, g_n$ and a sos polynomial h such that

$$g(x) \cdot \left(\sum_{i=1}^{n} x_i - r \right) + \sum_{i=1}^{n} g_i(x) \cdot (x_i^2 - x_i) = 1 + h(x) . \tag{29}$$

▶ **Theorem C.1** (Grigoriev [12]). *Let $0 \leq k \leq (n-3)/2$ be an integer. If $k < r < n-k$, then any Positivstellensatz refutation of the knapsack system of equations with parameter r, as in Equation (29), has degree at least $2k+4$.*

Proof. Grigoriev constructs a functional $\mathcal{G}_r : \mathbb{R}[x_1, \ldots, x_n] \to \mathbb{R}$ such that: when $\mathcal{G}_r$ is applied to the left-hand side of Equation (29) it evaluates to 0, provided that the total degree of the left-hand side is at most n; and when $\mathcal{G}_r$ is applied to the right-hand side of Equation (29) it is at least 1, provided the total degree of the right-hand side is at most $2k+2$. This leads to a contradiction, hence constructing such a functional $\mathcal{G}_r$ suffices to prove that a Positivstellensatz refutation must have degree at least $\min\{n, 2k+4\}$. The theorem then follows, with the additional observation that if $\min\{n, 2k+4\} = n$ is odd, then we can actually obtain a lower bound of $n+1$ (since any sum-of-squares polynomial must have even degree).

The functional $\mathcal{G}_r$ is first defined on the quotient ring $\mathcal{A} = \mathbb{R}[x_1, \ldots, x_n]/\langle x_1^2 - x_1, \ldots, x_n^2 - x_n \rangle$. For $p \in \mathcal{A}$ define

$$\mathcal{G}_r(p) = \mathrm{Sym}^{uni}(p)(r).$$

In other words, $\mathcal{G}_r$ looks at the univariate polynomial formed from the symmetrization of p over the symmetric group, and evaluates it at the point r. Explicitly, for a monomial $x_S = \prod_{i \in S} x_i$ with $|S| = t$ we see by Lemma B.7 that $\mathcal{G}_r(x_S) = B_t$ where

$$B_t = \frac{r(r-1)\cdots(r-t+1)}{n(n-1)\cdots(n-t+1)}\,. \tag{30}$$

For $p \in \mathbb{R}[x_1, \ldots, x_n]$ let $\bar{p}$ be its canonical multilinear representative in $\mathcal{A}$. The definition of the functional $\mathcal{G}_r$ is extended from $\mathcal{A}$ to the polynomial ring by letting $\mathcal{G}_r(p) := \mathcal{G}_r(\bar{p})$ for $p \in \mathbb{R}[x_1, \ldots, x_n]$.

Grigoriev's theorem now follows from the following four observations about $\mathcal{G}_r$:

1. $\mathcal{G}_r\left(g(x) \cdot (\sum_i x_i - r)\right) = 0$ for all polynomials g with $\deg(g) < n$. It suffices to show this for $g(x) = x_S = \prod_{i \in S} x_i$ for some $S \subsetneq [n]$ with $|S| = t < n$. In this case, by Equation (30), $\mathcal{G}_r(x_S(\sum_i x_i - r)) = (n-t)B_{t+1} + (t-r)B_t = 0$.
2. $\mathcal{G}_r\left(g_i(x)(x_i^2 - x_i)\right) = 0$ for all polynomials g_i. This is because the canonical multilinear representative of $g_i(x)(x_i^2 - x_i)$ in the quotient ring $\mathcal{A}$ is the constant-0 polynomial, and $\mathrm{Sym}^{uni}(0)(r) = 0$.
3. $\mathcal{G}_r(1) = 1$ for all values of r. The symmetrization of the constant-1 polynomial is itself, and the constant-1 polynomial always evaluates to 1.
4. $\mathcal{G}_r\left(p^2(x)\right) \geq 0$ if p is a polynomial of degree at most $k+1$. By Blekherman's Theorem 2.3, if $p \in \mathcal{A}$ and $d = \deg(p)$ then

$$\begin{aligned}\mathrm{Sym}^{uni}(p^2)(x) = q_d(x) + x(n-x)q_{d-1}(x) + x(x-1)(n-x)(n-1-x)q_{d-2}(x) + \cdots \\ + x(x-1)\cdots(x-d+1)(n-x)(n-1-x)\cdots(n-d+1-x)q_0(r)\,.\end{aligned} \tag{31}$$

   It follows that $\mathrm{Sym}^{uni}(p^2)(x) \geq 0$ for $x \in [d-1, n-d+1]$. Thus if $k < r < n-k$, then $\mathcal{G}_r(p^2) \geq 0$ for any p of degree $\leq k+1$. By linearity this extends to any h that is a sum of squares of polynomials of degree $\leq k+1$.

The first two observations imply that the left-hand side of Equation (29) evaluates to 0 under $\mathcal{G}_r$ (provided the total degree of the left-hand side is at most n), while the last two observations imply that the right-hand side evaluates to at least 1 (provided the total degree on the right is at most $2k+2$). ◀

D Application of Grigoriev's bound to ℓ_∞-error sos degree

Let n be odd and let $f = f_{\lfloor n/2 \rfloor}$, that is $f(x) = (|x| - n/2)^2 - 1/4$. Our Theorem 1.1 gives that any sos polynomial approximating f with ℓ_∞-error at most $1/50$, needs degree $\Omega(n)$. The functional $\mathcal{G}_r$ defined by Grigoriev (see discussion above Equation (30)) can be used to show an incomparable result: any sos polynomial of degree $(n-1)/2$ has error at least $\Omega(1/\log n)$ in approximating f in ℓ_∞-norm.

▶ **Theorem D.1.** *Let n be odd and $f : \{0,1\}^n \to \mathbb{R}$ be defined as $f(x) = (|x| - n/2)^2 - 1/4$. Any sos polynomial of degree $(n-1)/2$ has error at least*

$$(1 - O(1/n)) \frac{\pi}{4} \frac{1}{\ln((n+1)/2) + \gamma + \ln(16)}$$

in approximating f in ℓ_∞ norm. Here $\gamma \approx 0.577$ is the Euler-Mascheroni constant.

Proof. Let $h : \{0,1\}^n \to \mathbb{R}$ be a sos polynomial of degree $(n-1)/2$ approximating f with ℓ_∞-error ϵ. Write $h(x) = f(x) + e(x)$ where e is the function of "errors" satisfying $|e(x)| \leq \epsilon$ for all $x \in \{0,1\}^n$. Let $\delta_y : \{0,1\}^n \to \{0,1\}$ be the delta function on the boolean cube, where $\delta_y(x) = 1$ if and only if $x = y$. Recall that $\mathcal{G}_{n/2}(f) = \mathrm{Sym}^{uni}(f)(n/2) = (n/2 - n/2)^2 - 1/4 = -1/4$. By linearity of $\mathcal{G}_{n/2}$ we have

$$\begin{aligned} \mathcal{G}_{n/2}(f+e) = -1/4 + \mathcal{G}_{n/2}(e) &= -1/4 + \mathcal{G}_{n/2}\left(\sum_{y \in \{0,1\}^n} e(y)\delta_y\right) \\ &\leq -1/4 + \epsilon \sum_{y \in \{0,1\}^n} |\mathcal{G}_{n/2}(\delta_y)| \,. \end{aligned}$$

On the other hand, $\mathcal{G}_{n/2}(f+e) \geq 0$ as $f+e$ is a sum-of-squares of polynomials of degree at most $(n-1)/2$ (property 4 in the proof of Theorem C.1). Thus

$$\epsilon \geq \left(4 \sum_{y \in \{0,1\}^n} |\mathcal{G}_{n/2}(\delta_y)|\right)^{-1} . \tag{32}$$

The main part of the proof will be to evaluate this sum.

Let $L_i : \mathbb{R} \to \mathbb{R}$ be the degree-n polynomial uniquely defined by

$$L_i(z) = \begin{cases} 1 & z = i \\ 0 & z \in \{0,1,2,\ldots,n\} \setminus \{i\} \end{cases} .$$

Then we see that $\mathcal{G}_{n/2}(\delta_y) = L_{|y|}(n/2)/\binom{n}{|y|}$, and so

$$\sum_{y \in \{0,1\}^n} |\mathcal{G}_{n/2}(\delta_y)| = \sum_{k=0}^{n} |L_k(n/2)| \,.$$

To do this sum, let us first simplify the summand

$$\begin{aligned}
|L_k(n/2)| &= \frac{\prod_{a=0,a\neq k}^{n} |n/2-a|}{\prod_{a=0,a\neq k}^{n} |k-a|} \\
&= \frac{\prod_{a=0}^{n} |n/2-a|}{k!(n-k)!|n/2-k|} \\
&= \frac{1}{2^{n+1}} \frac{n!!n!!}{k!(n-k)!|n/2-k|} \\
&= \frac{n!}{2^{2n-1}\left(\frac{n-1}{2}\right)!^2} \binom{n}{k} \frac{1}{|n-2k|} \\
&= \frac{n}{2^{2n-1}} \binom{n-1}{(n-1)/2} \binom{n}{k} \frac{1}{|n-2k|} ,
\end{aligned}$$

where $n!!$ is defined as $\prod_{j=0}^{\lceil n/2\rceil-1}(n-2j)$, and we used $n! = n!! \cdot 2^{(n-1)/2} \cdot ((n-1)/2)!$ for odd n in the penultimate equality.

For what comes next, it will be more convenient to express $|L_k(n/2)|$ in terms of $m = (n-1)/2$. In this way, we obtain an expression defined for all m, rather than just odd n.

$$|L_k(m+1/2)| = \frac{2m+1}{2^{4m+1}} \binom{2m}{m} \binom{2m+1}{k} \frac{1}{|2m-2k+1|}$$

Let $A(m)$ denote the sum over $k = 0, \ldots, n = 2m+1$, which is

$$A(m) = \frac{2m+1}{2^{4m+1}} \binom{2m}{m} \sum_{k=0}^{2m+1} \frac{1}{|2m-2k+1|} \binom{2m+1}{k} .$$

By symmetry of the binomial coefficients we can multiply by 2 and sum over only half of them, thereby removing the absolute values.

$$A(m) = \frac{2m+1}{4^{2m}} \binom{2m}{m} \sum_{\ell=0}^{m} \frac{1}{2\ell+1} \binom{2m+1}{\ell+m+1}$$

Now we look at the difference between consecutive $A(m)$:

▶ **Claim D.2.**

$$A(m+1) - A(m) = \left(\frac{\binom{2(m+1)}{m+1}}{4^{m+1}} \right)^2$$

Proof. It is somewhat cumbersome to verify this claim directly. We take the following approach. Let $A(m) = B(m)C(m)$, where

$$B(m) = \frac{2m+1}{4^{2m}} \binom{2m}{m}, \qquad C(m) = \sum_{k=0}^{m} \frac{1}{2k+1} \binom{2m+1}{k+m+1} .$$

Note that

$$\frac{B(m+1)}{B(m)} = \frac{2m+3}{8(m+1)} .$$

Since $B(0) = 1$, this resolves to

$$B(m+1) = \frac{(2m+3)!!}{8^{m+1}(m+1)!} = \frac{2m+3}{4^{2(m+1)}} \binom{2(m+1)}{m+1} .$$

By Zeilberger's algorithm [26] we find a recurrence satisfied by the summand of $C(m)$.

$$\frac{2m+3}{2k+1}\binom{2m+3}{k+m+2} - \frac{8(m+1)}{2k+1}\binom{2m+1}{k+m+1} = \binom{2(m+1)}{m+k+1} - \binom{2(m+1)}{m+k+2}.$$

Summing this recurrence over $k = 0, \ldots, m+1$ we find

$$(2m+3)C(m+1) - 8(m+1)C(m) = \binom{2(m+1)}{m+1}.$$

This means that

$$A(m+1) - \underbrace{\frac{8(m+1)}{2m+3}B(m+1)}_{B(m)}C(m) = \frac{B(m+1)}{2m+3}\binom{2(m+1)}{m+1},$$

and in turn

$$A(m+1) - A(m) = \frac{1}{4^{2(m+1)}}\binom{2(m+1)}{m+1}^2.$$

◀

As $A(0) = 1$ this gives

$$A(m) = \sum_{i=0}^{m}\left(\frac{\binom{2i}{i}}{4^i}\right)^2.$$

Luckily, the latter sum has already been asymptotically evaluated in the study of the quantum adversary bound for the ordered search problem [7]. There it is shown that

$$\sum_{i=0}^{N}\left(\frac{\binom{2i}{i}}{4^i}\right)^2 = \frac{1}{\pi}\left(\ln(N+1) + \gamma + \ln(16)\right) + O(1/N),$$

where $\gamma \approx 0.577$ is the Euler-Mascheroni constant.

This gives

$$\begin{aligned}\sum_{y\in\{0,1\}^n}|\mathcal{G}_{n/2}(\delta_y)| = A((n-1)/2) &= \sum_{i=0}^{(n-1)/2}\left(\frac{\binom{2i}{i}}{4^i}\right)^2 \\ &= \frac{1}{\pi}\left(\ln((n+1)/2) + \gamma + \ln(16)\right) + O(1/n).\end{aligned}$$

Plugging this into Equation (32) gives the theorem. ◀

Limits of Minimum Circuit Size Problem as Oracle

Shuichi Hirahara[1] **and Osamu Watanabe**[2]

1 **Department of Computer Science, The University of Tokyo, Tokyo, Japan**
hirahara@is.s.u-tokyo.ac.jp

2 **Department of Mathematical and Computing Science, Tokyo Institute of Technology, Tokyo, Japan**
watanabe@is.titech.ac.jp

Abstract

The Minimum Circuit Size Problem (MCSP) is known to be hard for statistical zero knowledge via a BPP-Turing reduction (Allender and Das, 2014), whereas establishing NP-hardness of MCSP via a polynomial-time many-one reduction is difficult (Murray and Williams, 2015) in the sense that it implies ZPP $\neq$ EXP, which is a major open problem in computational complexity.

In this paper, we provide strong evidence that current techniques cannot establish NP-hardness of MCSP, even under polynomial-time Turing reductions or randomized reductions: Specifically, we introduce the notion of *oracle-independent reduction* to MCSP, which captures all the currently known reductions. We say that a reduction to MCSP is oracle-independent if the reduction can be generalized to a reduction to MCSP^A for any oracle A, where MCSP^A denotes an oracle version of MCSP. We prove that no language outside P is reducible to MCSP via an oracle-independent polynomial-time Turing reduction. We also show that the class of languages reducible to MCSP via an oracle-independent randomized reduction that makes at most one query is contained in AM $\cap$ coAM. Thus, NP-hardness of MCSP cannot be established via such oracle-independent reductions unless the polynomial hierarchy collapses.

We also extend the previous results to the case of more general reductions: We prove that establishing NP-hardness of MCSP via a polynomial-time *nonadaptive* reduction implies ZPP $\neq$ EXP, and that establishing NP-hardness of *approximating circuit complexity* via a polynomial-time *Turing* reduction also implies ZPP $\neq$ EXP. Along the way, we prove that approximating Levin's Kolmogorov complexity is *provably* not EXP-hard under polynomial-time Turing reductions, which is of independent interest.

1998 ACM Subject Classification F.1.3 Complexity Measures and Classes

Keywords and phrases minimum circuit size problem, NP-completeness, randomized reductions, resource-bounded Kolmogorov complexity, Turing reductions

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.18

1 Introduction

The Minimum Circuit Size Problem (MCSP) asks, given a truth-table $T \in \{0,1\}^{2^n}$ and a size-parameter s, whether there exists a circuit on n variables of size at most s whose truth-table is T. Although it is easy to see that MCSP is in NP, MCSP is not known to be NP-hard.

MCSP is closely related to circuit complexity by its definition, and hence it is one of the central problems in computational complexity. There are a number of formal connections from the complexity of MCSP to important open problems of computational complexity: for example, if MCSP $\in$ P then $\mathsf{EXP}^{\mathsf{NP}} \not\subseteq \mathsf{P/poly}$ [14]; if MCSP $\in$ coNP then MA can

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 18; pp. 18:1–18:20

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

be derandomized ($\mathsf{MA} = \mathsf{NP}$) [1]. Therefore, it is important to determine the structural complexity of MCSP.

While there is substantial evidence that MCSP is not tractable in the sense that $\mathsf{MCSP} \notin \mathsf{BPP}$, it remains open whether MCSP is the hardest problem in NP, that is, NP-hard or not. In this paper, we will discuss why it is so difficult to establish NP-hardness of MCSP. We note that, when discussing relative hardness of a problem, there are several types of reductions. Our main focus will be general and powerful reductions such as polynomial-time Turing reductions and randomized reductions. That is, what problems (*e.g.,* SAT) can we solve by using MCSP as an oracle?

1.1 Background

In the seminal paper by Kabanets and Cai [14], on one hand, they exhibited evidence that MCSP is intractable; namely, they proved that factoring Blum integers can be solved faster than any known algorithms, assuming that $\mathsf{MCSP} \in \mathsf{P}$. On the other hand, they also proved that establishing NP-hardness of MCSP is difficult: if MCSP is NP-hard under a certain type of restricted polynomial-time reductions, then some circuit lower bounds hold (and, in particular, $\mathsf{EXP} \not\subseteq \mathsf{P/poly}$); thus, establishing NP-hardness of MCSP (under the restricted reductions) is at least as difficult as proving $\mathsf{EXP} \not\subseteq \mathsf{P/poly}$. To summarize, MCSP is "harder" than factoring Blum integers, whereas establishing NP-hardness is difficult.

These two sides have been significantly pushed forward. On the positive side on hardness of MCSP, Allender, Buhrman, Koucký, van Melkebeek and Ronneburger [1] proved cryptographic problems, such as the discrete logarithm problem and integer factoring, can be solved in $\mathsf{BPP}^{\mathsf{MCSP}}$ (*i.e.,* these problems reduce to MCSP under BPP-Turing reductions). Allender and Das [2] strengthened these results by showing that every language in statistical zero knowledge is in $\mathsf{BPP}^{\mathsf{MCSP}}$.

The negative side on hardness of MCSP was considerably strengthened by Murray and Williams [17]. They showed that, if MCSP is NP-hard under polynomial-time many-one reductions, then $\mathsf{EXP} \neq \mathsf{NP} \cap \mathsf{P/poly}$ (and, in particular, $\mathsf{EXP} \neq \mathsf{ZPP}$), which is one of the central open problems in computational complexity. Thus, it is difficult to establish NP-hardness of MCSP under (general) polynomial-time many-one reductions. Moreover, they showed that, under local reductions (*i.e.,* that cannot look at a whole input), MCSP is *provably* not hard even for PARITY. Allender, Holden, and Kabanets [4] showed similar results for an oracle version of MCSP. For example, they showed that PSPACE is provably not reducible to $\mathsf{MCSP}^{\mathsf{QBF}}$ via a log space reduction; here, for an oracle A, MCSP^A denotes a problem of asking the smallest size of a circuit with A-oracle gates.

Thus, the current status of our understanding of MCSP is as follows: under the restricted reductions (*e.g.,* local reductions), MCSP is not "hard" at all, which suggests that such restricted reductions are insufficient to discuss the relative hardness of MCSP; under polynomial-time many-one reductions, it is difficult to establish NP-hardness of MCSP; nevertheless, BPP-Turing reductions to MCSP are powerful enough to solve every problem in statistical zero knowledge.

Therefore, it is very interesting to investigate whether one can push the positive side and establish NP-hardness of MCSP, or else the negative side can be pushed: More specifically, can we prove NP-hardness of MCSP under general reductions, such as BPP reductions? Can we extend the results of Murray and Williams [17] (as well as [4]) to more general reductions?

1.2 Oracle-independent Reductions

In this paper, we push the negative side further, and show that current techniques cannot be easily extended to show NP-hardness of MCSP. Specifically, we observe that current techniques do not rely on any inherent property of MCSP and instead rely on common properties that MCSP^A shares for an arbitrary oracle A. We thus introduce the notion of *oracle-independent reductions* to MCSP and then give upper bounds on classes of languages that reduces to MCSP via such reductions. We say that a reduction to MCSP is *oracle-independent* if the reduction can be generalized to MCSP^A for an arbitrary oracle A. In other words, the reduction exploits only properties common to MCSP^A for any oracle A (instead of unrelativizing properties of MCSP).

All the known efficient reductions to MCSP are oracle-independent. The main ingredient used by almost all the reductions [1, 2] is the construction from a one-way function to a pseudorandom generator by Håstad, Impagliazzo, Levin, and Luby [12]: Specifically, since the output of a pseudorandom (function) generator is efficiently computable, the output regarded as a truth-table has significantly low circuit complexity, compared to that of a truth-table chosen from a uniform distribution. Thus, MCSP constitutes a statistical test that distinguishes a pseudorandom distribution from a uniform distribution, which enables us to break a one-way function on average, thanks to [12]. This argument exploits only the fact that MCSP constitutes a statistical test. It is easy to see that an oracle version MCSP^A can also constitute a statistical test, and hence such reductions are oracle-independent.

Recently, new types of reductions to MCSP that do not rely on breaking a one-way function have been developed by Allender, Grochow, and Moore [3]. Based on new ideas, they showed that a certain graph isomorphism problem is reducible to MCSP via a randomized reduction with zero-sided error. We will see that their reductions are also oracle-independent.

A high-level reason why these reductions are oracle-independent is as follows: We are prone to rely on the fact that a randomly chosen truth-table requires high circuit complexity, because it is in general difficult to obtain a circuit lower bound on an explicit function. The fact that many truth-tables require high circuit complexity remains unchanged for any oracle version MCSP^A, and hence a reduction that only exploits this fact (as a circuit lower bound) is inevitably oracle-independent.

We provide strong evidence that NP-hardness of MCSP cannot be shown via such oracle-independent reductions. For deterministic reductions, we prove that nothing interesting is reducible to MCSP via an oracle-independent reduction:

▶ **Theorem 1.1.** *No language outside* P *can reduce to* MCSP *under polynomial-time Turing oracle-independent reductions. In other words, if a language L polynomial-time-Turing-reduces to MCSP^A for any oracle A, then $L \in \mathsf{P}$; it can be also simply stated as*

$$\bigcap_A \mathsf{P}^{\mathsf{MCSP}^A} = \mathsf{P}.$$

In contrast to previous work [14, 17, 4] which shows that NP-hardness of MCSP implies surprising consequences (*e.g.,* $\mathsf{EXP} \not\subseteq \mathsf{P/poly}$), we emphasize that this theorem gives us an inherent limitation of a deterministic oracle-independent reduction. One implication is that NP-hardness of MCSP cannot be shown via a deterministic oracle-independent reduction unless $\mathsf{P} = \mathsf{NP}$.

We note that this precisely captures the limit of what we can deterministically reduce to MCSP. Indeed, currently no (nontrivial) deterministic reduction to MCSP is known at all. The theorem suggests one reason behind this fact: in order to construct a deterministic reduction to MCSP, we need to use a property of MCSP that cannot be generalized to MCSP^A

for all A, which appears very difficult due to our few knowledge about nonrelativizing circuit lower bounds.

It should be also noted[1] that Theorem 1.1 implies that there exists an oracle A such that $\mathsf{MCSP} \not\leq_T^p \mathsf{MCSP}^A$ (unless $\mathsf{MCSP} \in \mathsf{P}$). At first glance (mainly due to its notation), it might be counterintuitive that an oracle version MCSP^A becomes "easier" than MCSP. The point is that the oracle A in the notation MCSP^A refers to the fact that a circuit that is minimized has oracle access to A, but this does not necessarily increase the computational difficulty of minimizing such an A-oracle circuit.

Indeed, we exploit this fact to prove Theorem 1.1. Roughly speaking, for any oracle-independent reduction to MCSP, we adversarially choose an oracle A so that any query that the reduction makes has circuit complexity of $O(\log n)$. Specifically, let $T_1, \ldots, T_{n^{O(1)}}$ be the truth-tables queried by the reduction (on some computation path); we encode these truth-tables into A so that the truth-table of $A(i, \text{-})$ is equal to T_i for any i. For this oracle, the reduction cannot query any truth-table that has high circuit complexity (relative to oracle A) because the size of the circuit that outputs $A(i, x)$ on input x is $O(\log n)$ for any i. We then simulate the reduction by exhaustively[2] search small circuits of size up to $O(\log n)$.

We also prove that even randomized oracle-independent reduction is not sufficient to establish NP-hardness of MCSP:

▶ **Theorem 1.2.** *If a language L is reducible to MCSP via an oracle-independent randomized reduction with negligible error that makes at most one query, then $L \in \mathsf{AM} \cap \mathsf{coAM}$. In other words,*

$$\bigcap_A \mathsf{BPP}^{\mathsf{MCSP}^A[1]} \subseteq \mathsf{AM} \cap \mathsf{coAM}.$$

Here, $\mathsf{BPP}^{B[1]}$ denotes the class of languages reducible to an oracle B via a randomized reduction with negligible error that makes at most one query.

In particular, $\bigcap_A \mathsf{BPP}^{\mathsf{MCSP}^A[1]}$ does not contain NP unless $\mathsf{NP} \subseteq \mathsf{coAM}$ (and in particular the polynomial hierarchy collapses [7]). Therefore, it is impossible to establish NP-hardness of MCSP via such reductions (unless the polynomial hierarchy collapses).

Oracle-independent Reductions vs. Relativization

We note that an oracle-independent reduction is different from simple relativization. In a relativization setting, Ko [15] showed the existence of a relativized world where MCSP is an NP-intermediate problem: MCSP is neither in coNP nor is NP-complete under polynomial-time Turing reductions. Specifically, he constructed an oracle A such that NP^A is not contained in $\mathsf{P}^{\mathsf{MCSP}^A, A}$, thereby showing a relativized world where MCSP cannot be NP-hard under polynomial-time Turing reductions. This shows the computational limit of MCSP in a relativized world.

In contrast, we discuss the computational limit of MCSP in a *real world* when MCSP is used by oracle-independent reductions. Technically, by exploiting the fact that NP-machines have an oracle access, Ko [15] constructed an oracle A so that some NP^A-computation

[1] This observation was given by one of the referees of CCC 2016 in the review report.

[2] When the "size" of a circuit refers to the number of its wires, we cannot enumerate all such circuits in polynomial time since there are $O(\log n)^{O(\log n)} = n^{O(\log \log n)}$ possible circuits of size less than $O(\log n)$, which gives only a weak upper bound. We will thus regard the "size" of a circuit as its description length, and also require that we can encode a truth-table into an oracle efficiently.

would go beyond the class $\mathsf{P}^{\mathsf{MCSP}^A, A}$. On the other hand, we construct an oracle A so that $\mathsf{P}^{\mathsf{MCSP}^A}$-computation cannot be strong; in fact, it is essentially the same as P.

1.3 Reductions to MCSP Imply Separations of Complexity Classes

We also extend the results of Murray and Williams [17] to the case of polynomial-time nonadaptive reductions and polynomial-time Turing reductions. In the former case, we prove that the same (in fact, slightly stronger) consequence can be obtained:

▶ **Theorem 1.3.** *It holds that* $\mathsf{P}_{||}^{\mathsf{MCSP}} \cap \mathsf{P/poly} \neq \mathsf{EXP}$ *(unconditionally). As a consequence, if* MCSP *is* NP*-hard via a polynomial-time nonadaptive reduction, then* $\mathsf{P}_{||}^{\mathsf{NP}} \cap \mathsf{P/poly} \neq \mathsf{EXP}$.

Here, $\mathsf{P}_{||}^{\mathsf{MCSP}}$ denotes the class of languages reducible to MCSP via a polynomial-time nonadaptive reduction.

Our proof is based on the firm links between circuit complexity and resource-bounded Kolmogorov complexity, which were established by a line of work [1, 5]. In fact, the proof is so simple that we can include a proof sketch here: Allender, Koucký, Ronneburger and Roy [5] showed that Levin's Kolmogorov complexity [16] (denoted by Kt) is polynomially related to circuit complexity if and only if $\mathsf{EXP} \subseteq \mathsf{P/poly}$; thus, assuming that $\mathsf{EXP} \subseteq \mathsf{P/poly}$, circuit complexity is essentially equal to Kt-complexity. Moreover, it is well-known that $\mathsf{EXP} \neq \mathsf{P}_{||}^{\mathrm{Kt}}$ (since a polynomial-time algorithm cannot output any strings of high Kt-complexity). Thus, assuming that $\mathsf{EXP} \subseteq \mathsf{P/poly}$, we also have $\mathsf{EXP} \neq \mathsf{P}_{||}^{\mathsf{MCSP}}$. This implies that $\mathsf{EXP} \neq \mathsf{P}_{||}^{\mathsf{MCSP}} \cap \mathsf{P/poly}$ (as otherwise we may assume $\mathsf{EXP} \subseteq \mathsf{P/poly}$). Therefore, at the core of the proof of the unconditional separation in Theorem 1.3 is $\mathsf{EXP} \neq \mathsf{P}_{||}^{\mathrm{Kt}}$.

Now we would like to extend the argument above into the case of polynomial-time Turing reductions. Unfortunately, we could not prove $\mathsf{EXP} \neq \mathsf{P}^{\mathrm{Kt}}$ (and this is an open problem since [1]). Nevertheless, we prove that a promise problem of approximating Kt within additive error $\omega(\log n)$ is not EXP-hard under polynomial-time Turing reductions, which is of independent interest:

▶ **Theorem 1.4.** *For any nondecreasing function* $g(n) = \omega(\log n)$*, let* $\mathrm{Gap}_g\mathrm{Kt}$ *denote a promise problem that asks for approximating* $\mathrm{Kt}(x)$ *within additive error* $g(|x|)$ *on input* x*. Then,* $\mathsf{EXP} \neq \mathsf{P}^{\mathrm{Gap}_g\mathrm{Kt}}$.

We note that, for a fixed exponential time $t(n) \geq 2^{n^2}$, Buhrman and Mayordomo [8] proved that K^t is not EXP-hard under polynomial-time Turing reductions. Here, K^t denotes resource-bounded Kolmogorov complexity such that a universal Turing machine that outputs x is required to run in time $t(|x|)$.

Now we can translate the property of Kt-complexity into that of MCSP, under the assumption that $\mathsf{EXP} \subseteq \mathsf{P/poly}$. As a consequence, we obtain:

▶ **Theorem 1.5.** *Let* $\mathrm{Gap}^k\mathrm{MCSP}$ *be a promise problem that asks for approximating the logarithm of circuit complexity within a factor of* k*. Then, there exists a constant* $k \geq 1$ *such that* $\mathsf{EXP} \neq \mathsf{P}^{\mathrm{Gap}^k\mathrm{MCSP}} \cap \mathsf{P/poly}$*. In particular, if a language* L *is reducible to* $\mathrm{Gap}^k\mathrm{MCSP}$ *via a polynomial-time Turing reduction for all* $k \geq 1$*, then* $\mathsf{P}^L \cap \mathsf{P/poly} \neq \mathsf{EXP}$.

In particular, establishing NP-hardness of $\mathrm{Gap}^k\mathrm{MCSP}$ via a polynomial-time Turing reduction requires separating $\mathsf{P}^{\mathsf{NP}} \cap \mathsf{P/poly}$ from EXP.

Interestingly, as observed in [5], the BPP-reductions of [1, 2] are extremely robust in terms of approximation. Specifically:

▶ **Theorem 1.6** (Analogous to [5, Theorem 19]). *For all $k \geq 1$, every language in statistical zero knowledge is reducible to* $\mathrm{Gap}^k\mathrm{MCSP}$ *via a* BPP*-Turing reduction.*

These two results exhibit a striking contrast between BPP-reductions and polynomial-time Turing reductions: BPP-reductions enable us to base hardness of approximating circuit complexity on hardness of statistical zero knowledge, whereas derandomizing the BPP-reduction requires a separation of complexity classes.

Organization

The rest of the paper is organized as follows. In Section 2, we introduce some notation and the definition of circuit complexity. In Section 3, we observe that the known reductions to MCSP are oracle-independent. We prove Theorem 1.1 in Section 4, and outline a proof of Theorem 1.2 in Section 5 (see the full version for the whole proof of Theorem 1.2). In Section 6, we extend the results of Murray and Williams [17] into the case of more general reductions.

2 Preliminaries

Since we need to specify an exact definition of circuit complexity in order to discuss some subtle details, we specify how to encode two strings into one string:

▶ **Definition 2.1.** For two strings $x, y \in \{0,1\}^*$, define the *pairing function* as $\langle x, y\rangle := 1^{|x|}0xy$.

We often write (x, y) instead of $(\langle x, y\rangle)$. We also abbreviate $\langle x, \langle y, z\rangle\rangle$ as $\langle x, y, z\rangle$. Note that $|\langle x, y\rangle| = 2|x| + |y| + 1$.

An oracle A is a subset of strings (*i.e.,* $A \subseteq \{0,1\}^*$). We identify a subset A of strings with its characteristic function $A\colon \{0,1\}^* \to \{0,1\}$. When we use diagonalization arguments, it is convenient to have the notion of *finite oracle*:

▶ **Definition 2.2.**

1. We say that A_0 is a *finite oracle* if $A_0\colon \{0,1\}^* \to \{0,1,\perp\}$ and $A_0(x) = \perp$ for all but finitely many strings $x \in \{0,1\}^*$, where $\perp$ means "undefined."
2. For an oracle $A \subseteq \{0,1\}^*$ and a finite oracle A_0, we say that A is *consistent* with A_0 if $A(x) = A_0(x)$ for any $x \in \{0,1\}^*$ such that $A_0(x) \neq \perp$.
3. Similarly, for $l \in \mathbb{N}$, we say that A and A_0 are *consistent up to length* l if it holds that $A(x) = 1$ if and only if $A_0(x) = 1$ for all strings $x \in \{0,1\}^*$ of length at most l.

For a nonnegative integer $n \in \mathbb{N}$, we write $[n] := \{1, \cdots, n\}$. For a string $x \in \{0,1\}^n$ and $i \in [n]$, we denote by x_i the ith bit of x. We also denote by $\underline{i}_n$ an integer i padded to length n. More specifically:

▶ **Definition 2.3.** For $n \in \mathbb{N}$ and $i \in [2^n]$, let $\underline{i}_n$ denote the ith string of $\{0,1\}^n$ in the lexicographic order.

For a set R, we write $r \in_R R$ to indicate that r is a random sample from the uniform distribution on R. For a distribution $\mathcal{D}$, we write $r \sim \mathcal{D}$ to indicate that r is a random sample from $\mathcal{D}$.

2.1 Definition of Circuit Size

Throughout this paper, we regard the description length of a circuit as its size. Thus, it is convenient to define the size of a circuit in terms of Kolmogorov complexity.

▶ **Definition 2.4.** Let U be a Turing machine. The *Kolmogorov complexity* $\mathrm{K}_U(x)$ of a string $x \in \{0,1\}^*$ with respect to U is defined as $\mathrm{K}_U(x) := \min\{\, |d| \mid U(d) = x \,\}$.

While we follow this standard definition, we use Kolmogorov complexity in a somewhat nonstandard way for discussing circuit complexity. We assume that a string x for which we consider its Kolmogorov complexity is a truth-table of a Boolean function. Thus, $|x|$ is 2^n for some $n \in \mathbb{N}$. We use a circuit interpreter for U instead of a universal Turing machine. In particular, for technical reasons, throughout this paper we will use a specific interpreter I that is defined below.

We first fix our standard (oracle) circuit interpreter. We assume any standard way to encode circuits by binary strings. Note that a circuit may be an oracle circuit that can use oracle gates outputting $A(z)$ for a given input z to the gate when a circuit is used with oracle A. Let I_0 denote a circuit interpreter for this encoding: that is, for any oracle A and a given description d of an oracle circuit C, the interpreter $I_0^A(d)$ yields the truth-table of C^A. (Thus, $|I_0^A(d)| = 2^n$ for some n and $I_0^A(d) = C^A(\underline{1}_n) \cdots C^A(\underline{2^n}_n)$.)

We will use the following facts that the standard circuit interpreter I_0^A should have:

1. $I_0^A(d)$ is computable in time polynomial in $|d|$ and $|I_0^A(d)|$, given oracle access to A.
2. For all but finitely many truth-tables $T \in \{0,1\}^*$ (where $|T|$ is a power of 2), there exists a circuit description of size less than $|T|^2$: that is, $\mathrm{K}_{I_0}(T) < |T|^2$.
3. Any oracle circuit C whose description length is at most m cannot query to an oracle any string of length greater than m. Thus, the output of C^A only depends on the membership in A of strings of length at most m.

We modify the standard circuit interpreter I_0 so that we can describe some type of circuits succinctly. For any $n \in \mathbb{N}$ and $d \in \{0,1\}^*$, let $C^A_{n,d}(x)$ be an oracle circuit that computes $A(x,d)$ (*i.e.,* $A(\langle x,d\rangle)$) for a given input $x \in \{0,1\}^n$, by using a single oracle gate with input $\langle x,d\rangle$.

▶ **Definition 2.5.** Define an interpreter I^A as follows:

$$I^A(0d) := I_0^A(d),$$
$$I^A(1^n,d) := I_0^A(C^A_{n,d}) = A(\underline{1}_n,d)A(\underline{2}_n,d)\cdots A(\underline{2^n}_n,d),$$

for any $n \geq 1$ and $d \in \{0,1\}^*$. For the other strings d (*e.g.,* $d = 1101$), leave $I^A(d)$ undefined.

For $A = \varnothing$, we write I instead of $I^{\varnothing}$.

▶ Remark.

1. Recall that $\langle 1^n,d\rangle = 1^n01^nd$; hence I^A is well-defined. Also, the definition of I^A ensures that the description length of a circuit $C^A_{n,d}$ is at most $|\,\langle 1^n,d\rangle\,| = 2n + |d| + 1$, which is exactly equal to the length of a query $\langle \underline{i}_n,d\rangle$ to oracle A.
2. For $A = \varnothing$, we have $\mathrm{K}_I(x) = \mathrm{K}_{I_0}(x) + 1$ for any $x \in \{0,1\}^* \setminus \{0\}^*$; hence, there is essentially no difference between our circuit complexity measure $\mathrm{K}_I(x)$ and a standard description length $\mathrm{K}_{I_0}(x)$. In particular, the results of Section 6 hold under any standard circuit complexity (*e.g.,* that counts the number of gates or wires).
3. For a general oracle A, since we assumed that the circuit $C^A_{n,d}$ can be described succinctly, we cannot guarantee that minimizing our complexity measure K_{I^A} is computationally equivalent to minimizing standard circuit complexity. However, all of the previous work (*e.g.,* [15, 4]) that we are aware of holds under our encoding scheme.

We define the minimum oracle circuit size problem MCSP^A by using I^A as a circuit interpreter:

▶ **Definition 2.6.** The *minimum oracle circuit size problem* MCSP^A relative to an oracle $A \subseteq \{0,1\}^*$ takes a truth-table $T \in \{0,1\}^*$ and a size-parameter $s \in \mathbb{N}$, and decides if $\mathrm{K}_{I^A}(T) \leq s$.

3 Why Are the Known Reductions Oracle-independent?

In this section, we argue that the known reductions to MCSP are oracle-independent. We observe that the existing reductions only exploit (as a circuit lower bound) the fact that many truth-tables require high (unrelativized) circuit complexity. Indeed, in the case of the reductions [1, 2] that rely on breaking a one-way function, the following holds:

▶ **Theorem 3.1** (Allender and Das [2]; see also [5, 1])**.** *Let $\epsilon \in (0,1)$ be a constant and let B be an oracle of polynomial density such that $\mathrm{K}_I(x) \geq |x|^\epsilon$ for any $x \in B$ (i.e., B is a statistical test that accepts "random" strings) . Then, every language in statistical zero knowledge is reducible to B via a BPP-reduction.*

Here, we say that an oracle B is of polynomial density if there exists a polynomial p such that $\Pr_{x \in_R \{0,1\}^n}[x \in B] \geq 1/p(n)$ for any $n \in \mathbb{N}$.

It is easy to see that such an oracle B can be computed, given oracle access to MCSP: indeed, define $B := \{\, x \in \{0,1\}^* \mid \mathrm{K}_I(x) \geq |x|^{1/2} \,\}$; it is obvious that $B \in \mathsf{P}^{\mathsf{MCSP}}$; moreover, since there are at most $2^{\sqrt{n}+1}$ strings that have circuit complexity at most $\sqrt{n}$ for any $n \in \mathbb{N}$, almost all strings of length n are in B. Therefore, every language in statistical zero knowledge is reducible to MCSP via a BPP-reduction.

This argument is still valid in the case of an oracle version MCSP^A: indeed, we may define an oracle B_A as $\{\, x \in \{0,1\}^* \mid \mathrm{K}_{I^A}(x) \geq |x|^{1/2} \,\}$ ($\in \mathsf{P}^{\mathsf{MCSP}^A}$); since $\mathrm{K}_I(x) \geq \mathrm{K}_{I^A}(x)$ for any $x \in \{0,1\}^*$, the hypothesis of the theorem remains satisfied.

Next, we show that an oracle-independent one-query reduction to MCSP allows us to convert a randomized algorithm with two-sided error into a randomized algorithm with zero-sided error. Moreover, the error probability is negligible.

▶ **Theorem 3.2** (Kabanets and Cai [14])**.** $\mathsf{BPP} \subseteq \bigcap_A \mathsf{ZPP}^{\mathsf{MCSP}^A[1]}$.

Proof Sketch. Pick a truth-table T uniformly at random. By making a query to MCSP^A, check if $\mathrm{K}_{I^A}(T) = n^{\Omega(1)}$. (Note that this also implies that $\mathrm{K}_I(T) = n^{\Omega(1)}$.) Now, if we successfully found a truth-table T that requires high circuit complexity, then we can use the pseudorandom generator by Impagliazzo and Wigderson [13] to derandomize a BPP computation. See [14] for the details. ◀

Finally, we observe that the new reductions by Allender, Grochow, and Moore [3] are oracle-independent. In fact, their reductions are not known to work under a usual definition of circuit size; instead, they presented reductions to *a* minimum circuit size problem, where "circuit size" here refers to KT-complexity. Let us recall KT-complexity briefly:

▶ **Definition 3.3** (KT-complexity [1])**.** Fix a universal (oracle) Turing machine U. For an oracle A, the KT^A-complexity of a string x is defined as

$$\mathrm{KT}^A(x) := \min\{\, |d| + t \mid \ U^{A,d}(i) = x_i \text{ in } t \text{ steps for all } i \in [|x|+1] \ \}.$$

Here, $x_{|x|+1}$ is defined as $\perp$ (a stop symbol).

It is known that KT^A-complexity is polynomially related to circuit complexity relative to A; hence, we may regard KT^A as a version of circuit complexity. In order to capture KT-complexity by our notation, we define a circuit interpreter I_0^A as follows: On input 1^t0d, run the universal Turing machine $U^{A,d}(i)$ for each $i \geq 1$ one by one in time at most t. Let n be the minimum i such that $U^{A,d}(i)$ outputs $\perp$. Output the concatenation of $U^{A,d}(1), \cdots, U^{A,d}(n-1)$. This definition ensures that $\mathrm{K}_{I_0^A}(x) = \mathrm{KT}^A(x) + 1$, and that $\mathrm{K}_{I^A}(x) \leq \mathrm{K}_{I_0^A}(x) + 1 = \mathrm{KT}^A(x) + 2$.

For this particular interpreter I^A, we prove:

▶ **Theorem 3.4** (Allender, Grochow, and Moore [3]). *For any oracle A, the rigid graph isomorphism problem is reducible to MCSP^A via a one-query BPP-reduction.*

Proof Sketch. We only observe why their reduction still works for MCSP^A, where A denotes an arbitrary oracle A. See [3] for the details.

Given two graphs (G_0, G_1), they constructed a string x' whose length is a power of 2 and a threshold θ that satisfy the following: If the graphs are isomorphic, then $\mathrm{KT}(x') \ll \theta$ with probability 1. If the graphs are rigid and not isomorphic, then x' contains information about a uniformly chosen random string of length at least θ, and hence $\mathrm{KT}(x') \geq \mathrm{K}_U(x') \gg \theta$ with high probability. (Here, $\mathrm{K}_U(x')$ denotes the *time-unbounded* Kolmogorov complexity.)

Now consider an arbitrary oracle A. We claim that the rigid graph isomorphism problem reduces to checking if $(x', \theta) \in \mathsf{MCSP}^A$. Suppose that the graphs are isomorphic; in this case, we have $\mathrm{K}_{I^A}(x') \leq \mathrm{KT}^A(x') + 2 \leq \mathrm{KT}(x') + 2 \ll \theta$. On the other hand, suppose that the graphs are rigid and not isomorphic. Since x' contains information about a uniformly chosen random string, an information-theoretic argument shows that $\mathrm{K}_{U^A}(x') \gg \theta$ with high probability (even relative to A). By the universality of U, we have $\mathrm{K}_{U^A}(x') \leq \mathrm{K}_{I^A}(x') + O(1)$. Therefore, $\mathrm{K}_{I^A}(x') \geq \mathrm{K}_{U^A}(x') - O(1) \gg \theta$. ◀

To summarize, on one hand, relativization does not increase circuit complexity ($\mathrm{K}_{I^A}(x') \leq \mathrm{K}_I(x')$); on the other hand, we are prone to rely on the fact that a uniformly chosen random string requires high circuit complexity, which remains true for any MCSP^A.

We mention that, for a specific oracle A, an efficient reduction to MCSP^A is known. Allender, Buhrman, Koucký, van Melkebeek and Ronneburger [1] showed that $\mathsf{PSPACE} \subseteq \mathsf{ZPP}^{\mathsf{MCSP}^{\mathsf{QBF}}}$. Since their proof relies on the fact that QBF is PSPACE-complete, the proof cannot be generalized to a reduction to MCSP; hence, their reduction cannot be regarded as an oracle-independent reduction to MCSP.

4 Limits of Oracle-independent Turing Reductions to MCSP

We show upper bounds for classes of languages that reduce to MCSP in an oracle-independent manner (*i.e.,* in a way that one does not use a property of MCSP rather than that of a relativized version MCSP^A). For example, we consider a situation where a language L is reducible to MCSP^A for any A via a polynomial-time Turing reduction; more precisely, for every A, there exists a polynomial-time Turing reduction from L to MCSP^A, *i.e.,* $L \in \bigcap_A \mathsf{P}^{\mathsf{MCSP}^A}$. That is, only properties common to MCSP^A for any oracle A are used to show that L is in $\mathsf{P}^{\mathsf{MCSP}^A}$. We would like to show that L is relatively easy in such situations.

In fact, we can indeed show that any language L in $\bigcap_A \mathsf{P}^{\mathsf{MCSP}^A}$ is in P.

▶ **Theorem 1.1** (restated). *Let $L \subseteq \{0,1\}^*$ be a language such that for any oracle A, there exists a polynomial-time Turing reduction from L to MCSP^A. Then L is in P. In short, $\bigcap_A \mathsf{P}^{\mathsf{MCSP}^A} = \mathsf{P}$.*

We will prove this theorem as follows: We will argue that, for each polynomial-time reduction M, we can adversarially choose an oracle A_M so that the reduction M cannot query any truth-table of high circuit complexity (by encoding the truth-tables queried by M into the oracle A_M). However, the assumption of the theorem states that a reduction M can depend on an oracle A, and hence A cannot depend on M. We first get around this difficulty by swapping the order of quantifiers: we reduce our theorem to the following lemma, in which a machine M cannot depend on A.

▶ **Lemma 4.1.** *Let $L \subseteq \{0,1\}^*$ be a language and A_0 be an arbitrary finite oracle. Suppose that there exists a polynomial-time oracle Turing machine M such that $M^{\mathsf{MCSP}^A}(x) = L(x)$ for any $x \in \{0,1\}^*$ and any oracle A consistent with A_0. Then, $L \in \mathsf{P}$.*

Note that, in this lemma, a *single* machine M is required to compute L with respect to *every* oracle version MCSP^A. We will later prove this lemma by choosing, for each reduction M and input x, an oracle $A_{M,x}$ so that the reduction M to $\mathsf{MCSP}^{A_{M,x}}$ can be simulated in polynomial time. Before its proof, we show that Lemma 4.1 implies Theorem 1.1 by using a simple diagonalization argument.

Proof of Theorem 1.1 based on Lemma 4.1. We prove the contraposition: Assuming $L \notin \mathsf{P}$, the aim is to construct an oracle A such that $L \notin \mathsf{P}^A$. Such an oracle $A = \bigcup_e B_e$ is constructed in stages. Let all the polynomial-time oracle Turing machines be $\{M_1, M_2, \cdots\}$.

At stage e, we construct a finite oracle B_e. At stage 0, set $B_0(y) := \bot$ for all $y \in \{0,1\}^*$. At stage $e \geq 1$, we apply Lemma 4.1 for $M = M_e$ and $A_0 = B_{e-1}$: by the assumption that $L \notin \mathsf{P}$, there exist some string x_e and some oracle B_e consistent with B_{e-1} such that $M_e^{\mathsf{MCSP}^{B_e}}(x_e) \neq L(x_e)$. We may assume that B_e is a finite oracle: indeed, since the computation of $M_e^{\mathsf{MCSP}^{B_e}}$ on input x_e makes a finite number of queries to MCSP^{B_e}, the answers of the queries also depend on a finite portion of B_e. Define an oracle A as the union of all the oracles B_e whose $\bot$ is replaced by 0.

Since A is consistent with B_e, it holds that $M_e^{\mathsf{MCSP}^{B_e}}(x_e) = M_e^{\mathsf{MCSP}^{A}}(x_e)$ for each $e \geq 1$. By the definition of x_e, we have $M_e^{\mathsf{MCSP}^{B_e}}(x_e) \neq L(x_e)$. Therefore, $M_e^{\mathsf{MCSP}^{A}}(x_e) \neq L(x_e)$ holds for any e, and hence $L \notin \mathsf{P}^{\mathsf{MCSP}^A}$. ◀

Now we give a proof of Lemma 4.1. The idea is as follows: For any reduction M and any input x, we simulate the reduction M by answering M's query by exhaustively searching all the circuits of size at most $O(\log n)$. On this specific computation path of M, we claim that there exists some oracle $A_{M,x}$ such that the simulated computation path coincides with the computation path of the reduction M to $\mathsf{MCSP}^{A_{M,x}}$, thereby showing that the output of the simulation of M is $L(x)$: Since M is a polynomial-time machine, the number of the queries on the computation path is at most $n^{O(1)}$. Thus, the index i of the queries can be described in $O(\log n)$ bits, and hence the description length of the oracle circuit $C^{A_{M,x}}(j) := A_{M,x}(j,i)$ is at most $O(\log n)$. By defining $A_{M,x}(j,i) := T_{ij}$ for each truth-table T_i queried by M, any truth-table T_i admits a circuit of size at most $O(\log n)$.

Let us turn to a formal proof. Let M be a polynomial-time oracle machine that computes L given oracle access to MCSP^A in time n^c for some constant c, where A denotes an arbitrary oracle consistent with A_0. We define a polynomial-time machine M_0 that simulates M without using MCSP^A as follows: On input $x \in \{0,1\}^*$ of length n, simulate M on input x, and accept if and only if M accepts. If M makes a query (T, s), then we try to compute the circuit complexity $\mathrm{K}_{I^{A_0}}(T)$ of the truth-table T relative to a finite oracle A_0, by an exhaustive search up to size at most $4c \log n$. (More specifically, we compute the shortest description d of length at most $4c \log n$ such that $I^{A_0}(d) = T$, where we regard $A_0 \subseteq \{0,1\}^*$ as an oracle by

replacing $\perp$ by 0 in finite oracle A_0.) If the circuit complexity $\mathrm{K}_{I^{A_0}}(T)$ has turned out to be greater than $4c\log n$, then define $s' := 4c\log n$; otherwise define $s' := \mathrm{K}_{I^{A_0}}(T)$ $(\leq 4c\log n)$. (*i.e.,* $s' := \min\{\, 4c\log n,\, \mathrm{K}_{I^{A_0}}(T)\,\}$.) Answer "Yes" to the query if and only if $s' \leq s$.

It is easy to see that M_0 is indeed a polynomial-time machine, since there are only $2^{O(\log n)}$ circuits of size at most $O(\log n)$. (Recall that we regard a circuit size as a description length.) Thus, it is sufficient to prove the following:

▶ **Claim 4.2.** *For all sufficiently large n and all inputs x of length n, there exists an oracle $A_{M,x}$ consistent with A_0 such that $M_0(x) = M^{\mathsf{MCSP}^{A_{M,x}}}(x)$.*

Note that the assumption of Lemma 4.1 implies that $M^{\mathsf{MCSP}^{A_{M,x}}}(x) = L(x)$. Thus, the claim implies that $M_0(x) = L(x)$ and hence $L \in \mathsf{P}$.

Proof of Claim 4.2. Fix n sufficiently large and an input $x \in \{0,1\}^n$. For $i \in [n^c]$, let T_i be the truth-table in the ith query that M makes on the computation path simulated by M_0 on input x.

We define an oracle $A_{M,x} = A$ as follows (here, $A_{M,x}$ is abbreviated as A for notational convenience): For any string $q \in \{0,1\}^*$ of length less than $4c\log n$, define $A(q) = 1$ if and only if $A_0(q) = 1$. For strings of length $4c\log n$, we encode T_i into oracle A so that the circuit complexity of T_i relative to A is at most $4c\log n$: Specifically, we would like to define a description d_i of length (*exactly* equal to) $4c\log n$ so that $I^A(d_i) = T_i$. To this end, let $a_i := \log |T_i|$ and define $d_i := \langle 1^{a_i}, \underline{i}_{k_i} \rangle$, where $k_i \in \mathbb{N}$ is defined so that $|d_i| = 2a_i + 1 + k_i = 4c\log n$. Here, $\underline{i}_{k_i}$ is well-defined: indeed, we have $a_i = \log |T_i| \leq c\log n$, which implies that $k_i := 4c\log n - 2a_i - 1 \geq c\log n$, and thus $i \leq 2^{c\log n} \leq 2^{k_i}$. Now define $A(\underline{j}_{a_i}, \underline{i}_{k_i}) := T_{ij}$ for each $j \in [2^{a_i}]$. By the definition of I^A, the truth-table T_i can be described succinctly: $I^A(d_i) = A(\underline{1}_{a_i}, \underline{i}_{k_i}) \cdots A(\underline{2^{a_i}}_{a_i}, \underline{i}_{k_i}) = T_i$; thus, the circuit complexity $\mathrm{K}_{I^A}(T_i)$ of T_i is at most $|d_i| = 4c\log n$.

It remains to show that, for each query (T_i, s) that M makes on the computation path simulated by M_0, circuit complexity s' $(= \min\{\, 4c\log n,\, \mathrm{K}_{I^{A_0}}(T_i)\,\})$ calculated by M_0 coincides with $\mathrm{K}_{I^A}(T_i)$; note that this implies that $M_0(x) = M^{\mathsf{MCSP}^A}(x)$, because the computation path simulated by M_0 coincides with that of M relative to MCSP^A. In order to see $\mathrm{K}_{I^A}(T_i) = \min\{\, 4c\log n,\, \mathrm{K}_{I^{A_0}}(T_i)\,\}$, first we note that A and A_0 are consistent up to length $4c\log n - 1$; thus, for small circuits, circuit complexity relative to A remains the same with circuit complexity relative to A_0, because small circuits cannot query long strings of length $4c\log n$. Formally, suppose that $\mathrm{K}_{I^{A_0}}(T_i) < 4c\log n$ (*i.e.,* $s' = \mathrm{K}_{I^{A_0}}(T_i)$). In this case, there exists some description d of length less than $4c\log n$ such that $I^{A_0}(d) = T_i$. Since the circuit described by d cannot make any query of length greater than $|d|$, it holds that $I^{A_0}(d) = I^A(d)$. Thus $\mathrm{K}_{I^A}(T_i) \leq \mathrm{K}_{I^{A_0}}(T_i) < 4c\log n$. Similarly, we have $\mathrm{K}_{I^{A_0}}(T_i) \leq \mathrm{K}_{I^A}(T_i)$, and hence $\mathrm{K}_{I^A}(T_i) = \mathrm{K}_{I^{A_0}}(T_i) = s'$. Now suppose that $\mathrm{K}_{I^{A_0}}(T_i) \geq 4c\log n$ (*i.e.,* $s' = 4c\log n$). We claim that $\mathrm{K}_{I^A}(T_i) = 4c\log n$. Since we have $\mathrm{K}_{I^A}(T_i) \leq 4c\log n$ by the definition of A, it is sufficient to show that $\mathrm{K}_{I^A}(T_i) < 4c\log n$ is not true. Assume, by way of contradiction, that $\mathrm{K}_{I^A}(T_i) < 4c\log n$. By the same argument above, it must be the case that $\mathrm{K}_{I^A}(T_i) \geq \mathrm{K}_{I^{A_0}}(T_i) \geq 4c\log n$, which is a contradiction. ◀

This completes the proof of Lemma 4.1.

▶ Remark. If we regard a size of a circuit as the number of its wires, then the upper bound P becomes $\mathsf{DTIME}(n^{O(\log\log n)})$. Specifically, let MCSP'^A denotes a version of MCSP^A in which a size of a circuit is measured by the number of its wires. Then we have $\bigcap_A \mathsf{P}^{\mathsf{MCSP}'^A} \subseteq \mathsf{DTIME}(n^{O(\log\log n)})$. This can be proved by simply changing M_0 in the proof above so that

M_0 exhaustively search all the circuits of at most $O(\log n)$ *wires* in time $O(\log n)^{O(\log n)} = n^{O(\log\log n)}$.

5 Limits of Oracle-independent Randomized Reductions to MCSP

In this section, we discuss the limits of a randomized reduction to MCSP that can be generalized to a reduction to MCSP^A for an arbitrary oracle A. Our focus is a randomized reduction with negligible two-sided error that can make at most one query:

▶ **Definition 5.1.** Let $L, B \subseteq \{0,1\}^*$ be a language and an oracle, respectively. We say that L reduces to B via a *one-query* BPP*-reduction* and write $L \in \mathsf{BPP}^{B[1]}$ if there exist polynomial-time machines M, Q and a negligible function ϵ such that, for any $x \in \{0,1\}^*$,

$$\Pr_{r\in\{0,1\}^{|x|^{O(1)}}}[M(x,r,B(Q(x,r))) = L(x)] \geq 1 - \epsilon(|x|).$$

Here, we say that a function ϵ is negligible if for all polynomials p, for all sufficiently large $n \in \mathbb{N}$, the function is bounded by the inverse of p: that is, $\epsilon(n) < \frac{1}{p(n)}$.

Note that we require the error probability to be negligible. Since the number of queries is restricted to one, we cannot apply the standard error-reduction argument; hence, this definition may be stronger than a definition whose error probability is a constant. We leave as an open problem improving our result to the case when the error probability is a constant.

We prove that there is no language outside $\mathsf{AM} \cap \mathsf{coAM}$ that can reduce to MCSP^A for an arbitrary oracle A via a one-query randomized reduction:

▶ **Theorem 1.2** (restated). *Let $L \subseteq \{0,1\}^*$ be a language such that for any oracle A, there exists a one-query* BPP*-reduction from L to MCSP^A. Then L is in $\mathsf{AM} \cap \mathsf{coAM}$. In short,*

$$\bigcap_A \mathsf{BPP}^{\mathsf{MCSP}^A[1]} \subseteq \mathsf{AM} \cap \mathsf{coAM}.$$

As with Theorem 1.1, we first swap the order of quantifiers. However, in order to swap the order of quantifiers, we need to enumerate all the negligible functions, which is not countably many; thus, we sidestep this by requiring that the error probability is an inverse polynomial $1/q$ in the running time of machines M and Q. Also, since a one-query BPP-reduction is closed under complement, we only have to show that the target language is in AM.

▶ **Lemma 5.2.** *There exists some universal polynomial q (specified later) that satisfies the following: Let L, A_0 be a language and a finite oracle, respectively. Suppose that there exist a polynomial p and Turing machines M, Q such that M and Q run in time $p(n)$ and*

$$\Pr_{r\in\{0,1\}^{p(n)}}[M(x,r,\mathsf{MCSP}^A(Q(x,r))) = L(x)] \geq 1 - \frac{1}{q(p(n))}$$

for any $x \in \{0,1\}^*$ of length n and any oracle $A \subseteq \{0,1\}^*$ consistent with A_0. Then, we have $L \in \mathsf{AM}$.

We prove that Lemma 5.2 implies Theorem 1.2:

Proof of Theorem 1.2 based on Lemma 5.2. We prove the contraposition: Assuming $L \notin \mathsf{AM}$, we will construct an oracle A such that $L \notin \mathsf{BPP}^{\mathsf{MCSP}^A[1]}$ by diagonalization.

Enumerate all the tuples $\{(M_e, Q_e, c_e)\}_{e\geq 1}$, where M_e and Q_e are polynomial-time machines and $c_e \in \mathbb{N}$. We assume that, for each tuple (M_e, Q_e, c_e), there exist infinitely many $e' \in \mathbb{N}$ such that $(M_e, Q_e, c_e) = (M_{e'}, Q_{e'}, c_{e'})$.

At stage $e \geq 1$, we construct a finite oracle B_e that fools a one-query BPP reduction (M_e, Q_e) that runs in time n^{c_e}: If M_e or Q_e does not run in time n^{c_e}, then we define $B_e := B_{e-1}$. Otherwise, we can apply the contraposition of Lemma 5.2 to M_e and Q_e: there exist some input x_e and some oracle B_e consistent with B_{e-1} such that $\Pr_r[M_e(x_e, r, \mathsf{MCSP}^{B_e}(Q_e(x_e, r))) = L(x_e)] < 1 - \frac{1}{q(n^{c_e})}$. We can make B_e a finite oracle, since M_e depends on only a finite portion of B_e. This completes stage e. Define A as the union of all the oracles B_e whose $\perp$ is replaced by 0.

We claim that $L \notin \mathsf{BPP}^{\mathsf{MCSP}^A[1]}$. Assume otherwise. Then there exist a constant $c > 1$, a negligible function ϵ, and machines M and Q that run in time n^c such that

$$\Pr_r[M(x, r, \mathsf{MCSP}^A(Q(x, r))) = L(x)] \geq 1 - \epsilon(|x|) \tag{1}$$

for all $x \in \{0,1\}^*$. Fix a sufficiently large $n_0 \in \mathbb{N}$ such that $\epsilon(n) < \frac{1}{q(n^{c+1})}$ for all $n \geq n_0$. Let M' be the Turing machine[3] that, on input x, outputs a hardwired answer $L(x)$ if $|x| \leq n_0$, and simulates M otherwise. Note that the running time of M' is at most n^{c+1}.

By the construction above, there exists $e \geq n_0$ such that $(M_e, Q_e, c_e) = (M', Q, c+1)$. By the definition of x_e, we have $\Pr_r[M'(x_e, r, \mathsf{MCSP}^A(Q(x_e, r))) = L(x_e)] < 1 - \frac{1}{q(|x_e|^{c+1})}$. Moreover, since M' outputs a correct answer with probability 1 on input x of length at most n_0, it holds that $|x_e| > n_0$; thus, we have $\epsilon(|x_e|) < \frac{1}{q(|x_e|^{c+1})}$; in addition, the machine M' behaves in the same way with M. Hence, the success probability of (M, Q) on input x_e is equal to that of (M', Q) on input x_e, which is bounded above by $1 - \frac{1}{q(|x_e|^{c+1})} < 1 - \epsilon(|x_e|)$. This contradicts (1). ◀

Now we outline the proof of Lemma 5.2.

We will first show that we may assume that all the queries that Q makes have a truth-table of a fixed length 2^t and a fixed size-parameter s for some $t, s \in \mathbb{N}$. There is no loss of generality in assuming this because there are only polynomially many possibilities: the number of all the possible lengths of a truth-table and size-parameters is at most n^c for some c. Moreover, we may fix how to use the answer of a query: specifically, for a random choice r, define $f\colon \{0,1\} \to \{0,1\}$ (which has 4 possible choices) so that $f(b) = M(x, r, b)$. (For example, $f(b) = b$ means that M accepts if and only if the query is a positive instance of MCSP^A.)

We classify the set of random choices r into $R_{f,t,s}$ according to these parameters (f, t, s). If $x \in L$, then there must exist some (f, t, s) such that $f(\mathsf{MCSP}^A(Q(x, r))) = 1$ with high probability over the choice of $r \in_R R_{f,t,s}$. On the other hand, if $x \notin L$, then any (f, t, s) must satisfy $f(\mathsf{MCSP}^A(Q(x, r))) = 0$ with high probability. Therefore, it is sufficient to prove that, for a specific (f, t, s), there exists an AM protocol that checks if $f(\mathsf{MCSP}^A(Q(x, r))) = 1$ with high probability conditioning on $r \in R_{f,t,s}$.

Let us assume that $f(b) = b$ for simplicity. Then, it is sufficient to estimate the probability

$$P_{f,t,s} := \Pr_{r \in_R R_{f,t,s}}[f(\mathsf{MCSP}^A(Q(x, r))) = 1] = \Pr_{r \in_R R_{f,t,s}}[Q(x, r) \in \mathsf{MCSP}^A]$$

by an AM protocol. If the probability $P_{f,t,s}$ is close to 1, then the distribution induced by $Q(x, r)$ concentrates on a limited number of instances: indeed, since there are at most 2^{s+1}

[3] M' can be implemented by a Turing machine as follows: Read the first $n_0 + 1$ bits of the input (if any). If the input length is at most n_0, then output the hardwired answer. Otherwise, move the head of the input tape to the initial position, and continue the computation of M. This implementation costs at most $2n_0$ additional steps.

positive instances in MCSP^A for a size-parameter s, the query $Q(x,r)$ must be one of such instances with probability at least $P_{f,t,s}$. Conversely, suppose that the query distribution $Q(x,r)$ concentrates on a limited number of instances $\{(T_1,s),(T_2,s),\cdots\}$; we may encode T_i into an oracle A and force these instances to be positive (*i.e.,* $(T_i,s)\in\mathsf{MCSP}^A$); as a result, the probability $P_{f,t,s}$ is not small (since the instances (T_i,s) are positive). Therefore, the task reduces to checking whether the query distribution concentrates on a limited number of instances.

To this end, we will use the heavy samples protocol [6]. We say that an instance (T,s) is β-heavy if the probability that (T,s) is queried (*i.e.,* $(T,s)=Q(x,r)$) is at least β. The heavy samples protocol allows us to estimate the probability that $Q(x,r)$ is β-heavy.

▶ **Lemma 5.3** (The heavy samples protocol; Trevisan and Bogdanov [6]). *Let $\mathcal{D}=\{\mathcal{D}_n\}_{n\in\mathbb{N}}$ be a polynomial-time samplable distribution. There exist a universal constant c ($c=2^{11}$ will do) and an $\mathsf{AM}\cap\mathsf{coAM}$ protocol that solves the following promise problem: Given input 1^n and a threshold $\beta\in[0,1]$, accept if $\Pr_{y\sim\mathcal{D}_n}[\,y \text{ is } c\beta\text{-heavy}\,]\geq\frac{3}{4}$, and reject if $\Pr_{y\sim\mathcal{D}_n}[\,y \text{ is } \beta\text{-heavy}\,]\leq\frac{1}{4}$.*

This lemma follows from the lower bound protocol (Goldwasser and Sipser [10]) and the upper bound protocol (Fortnow [9]).

Due to space constraints, we defer the formal proof to the full version.

6 Hardness of MCSP Implies Separations of Complexity Classes

In this section, we give a reinterpretation of the results of Murray and Williams [17] by using Levin's Kolmogorov complexity, and extend these results to the case of polynomial-time nonadaptive reductions and polynomial-time Turing reductions. Our proofs are based on the firm links between circuit complexity and resource-bounded Kolmogorov complexity, which have been established by a line of work [1, 5]. First, we introduce Levin's Kt-complexity.

▶ **Definition 6.1** (Levin's Kolmogorov Complexity [16]). Fix an efficient universal Turing machine U. The *Levin's Kolmogorov complexity* $\mathrm{Kt}(x)$ of a string x is defined as

$$\mathrm{Kt}(x):=\min\{\,|d|+\log t\mid U(d)\text{ outputs }x\text{ in time }t\,\}.$$

Our proof is principally based on the fact that $\mathsf{EXP}\subseteq\mathsf{P/poly}$ if and only if circuit complexity K_I is polynomially related to Levin's Kolmogorov complexity Kt.

▶ **Lemma 6.2** (Allender, Koucký, Ronneburger and Roy [5]). $\mathsf{EXP}\subseteq\mathsf{P/poly}$ *if and only if there exists a polynomial* poly *in two variables such that* $\mathrm{K}_I(x)\leq\mathsf{poly}(\mathrm{Kt}(x),\log|x|)$.

We would like to separate the class of languages reducible to MCSP from EXP, under the assumption that $\mathsf{EXP}\subseteq\mathsf{P/poly}$. Under this assumption, Lemma 6.2 suggests that circuit complexity and Kt-complexity are essentially the same (in the sense that these are polynomially related to each other). Therefore, we will first separate the class of languages reducible to Kt from EXP, and then, based on Lemma 6.2, translate the property of Kt into that of MCSP, assuming $\mathsf{EXP}\subseteq\mathsf{P/poly}$.

6.1 The Case of Nonadaptive Reductions

In the case of polynomial-time nonadaptive reductions, it is well known that $\mathsf{P}_{||}^{\mathrm{Kt}}\neq\mathsf{EXP}$.

▶ **Proposition 6.3** (folklore). $\mathsf{EXP}_{||}^{\mathrm{Kt}}=\mathsf{EXP}$. *(Here,* Kt *is identified with the oracle* $\{\,(x,s)\in\{0,1\}^*\times\mathbb{N}\mid\mathrm{Kt}(x)\leq s\,\}$.*)*

Note that this implies $\mathsf{P}_{||}^{\mathrm{Kt}} \neq \mathsf{EXP}$ by the time hierarchy theorem.

Proof. Let M be any $\mathsf{EXP}_{||}^{\mathrm{Kt}}$ machine. Given input $x \in \{0,1\}^*$ of length n, let $Q(x)$ be the set of queries (without size-parameter s) that M makes. Since M is a nonadaptive oracle machine, $Q(x)$ can be computed in exponential time. Therefore, any query $q \in Q(x)$ can be described by the input x and an index $i \in [2^{n^{O(1)}}]$ in exponential time; hence, $\mathrm{Kt}(q) \leq |x| + n^{O(1)} + \log 2^{n^{O(1)}} = n^{O(1)}$.

Given the fact that $\mathrm{Kt}(q) \leq n^{O(1)}$, we may compute $\mathrm{Kt}(q)$ by an exhaustive search in exponential time. Thus, by answering M's queries by the exhaustive search, we can compute M's output in exponential time. ◀

Under the assumption that $\mathsf{EXP} \subseteq \mathsf{P/poly}$, we can translate the property of Kt into that of circuit complexity:

▶ **Theorem 6.4.** *If* $\mathsf{EXP} \subseteq \mathsf{P/poly}$ *then* $\mathsf{EXP}_{||}^{\mathsf{MCSP}} = \mathsf{EXP}$.

Proof Sketch. Let (T, s) be any query of an $\mathsf{EXP}_{||}^{\mathsf{MCSP}}$ machine. Since $\mathrm{Kt}(T)$ is $n^{O(1)}$, the circuit complexity $\mathrm{K}_I(T)$ of T is also bounded above by $n^{O(1)}$ by Lemma 6.2. Thus, the circuit complexity of all the queries can be computed by an exhaustive search in time exponential in n. ◀

This theorem allows us to obtain a nontrivial separation of $\mathsf{P}_{||}^{\mathsf{MCSP}} \cap \mathsf{P/poly}$ from EXP:

▶ **Corollary 6.5.** $\mathsf{P}_{||}^{\mathsf{MCSP}} \cap \mathsf{P/poly} \neq \mathsf{EXP}$.

Proof. Assume, by way of contradiction, that $\mathsf{P}_{||}^{\mathsf{MCSP}} \cap \mathsf{P/poly} = \mathsf{EXP}$. In particular, $\mathsf{EXP} \subseteq \mathsf{P/poly}$. Thus, by Theorem 6.4, we have $\mathsf{EXP}_{||}^{\mathsf{MCSP}} = \mathsf{EXP}$. Therefore, $\mathsf{EXP}_{||}^{\mathsf{MCSP}} = \mathsf{EXP} = \mathsf{P}_{||}^{\mathsf{MCSP}}$, which contradicts the (relativized) time hierarchy theorem [11]. ◀

This result exhibits a singular property of MCSP. In particular, reducing a language L to MCSP via a polynomial-time nonadaptive reduction implies a separation of $\mathsf{P}_{||}^{L} \cap \mathsf{P/poly}$ from EXP.

▶ **Corollary 6.6.** *If* $L \leq_{tt}^{p} \mathsf{MCSP}$, *then* $\mathsf{P}_{||}^{L} \cap \mathsf{P/poly} \neq \mathsf{EXP}$.

Proof. The hypothesis implies that $\mathsf{P}_{||}^{L} \subseteq \mathsf{P}_{||}^{\mathsf{MCSP}}$, and by the previous corollary it holds that $\mathsf{EXP} \not\subseteq \mathsf{P}_{||}^{\mathsf{MCSP}} \cap \mathsf{P/poly}$, from which the result follows. ◀

We give some specific remarks:

▶ Remark.

1. If MCSP is ZPP-hard under polynomial-time nonadaptive reductions, then $\mathsf{ZPP} \neq \mathsf{EXP}$, which is a notorious open problem.
2. If MCSP is NP-complete under polynomial-time nonadaptive reductions, then $\mathsf{P}_{||}^{\mathsf{NP}} \cap \mathsf{P/poly} \neq \mathsf{EXP}$. (The consequence is also a tiny improvement of Murray and Williams [17], who showed that $\mathsf{NP} \cap \mathsf{P/poly} \neq \mathsf{EXP}$ under the assumption that $\mathsf{NP} \leq_{m}^{p} \mathsf{MCSP}$.)

6.2 On Hardness of Approximating Kt-complexity and Circuit Complexity

Now we turn to the case of polynomial-time Turing reductions. We first introduce some definitions about promise problems:

▶ **Definition 6.7.**
1. A *promise problem* $\Pi = (\Pi_Y, \Pi_N)$ is a pair of disjoint languages Π_Y and Π_N, where Π_Y is the set of YES instances and Π_N is the set of NO instances.
2. We say that an oracle A *satisfies the promise of* $\Pi = (\Pi_Y, \Pi_N)$ if, for any $x \in \{0,1\}^*$, it holds that $x \in \Pi_Y$ implies $x \in A$, and that $x \in \Pi_N$ implies $x \notin A$.
3. We say that a language L *is reducible to a promise problem* Π via a polynomial-time Turing reduction M and write $L \leq_T^p \Pi$ if $M^A(x) = L(x)$ for any $x \in \{0,1\}^*$ and any oracle A that satisfies the promise of Π.

We show that approximating Kt-complexity within additive error $g(n) = \omega(\log n)$ is not EXP-complete under polynomial-time Turing reductions. We denote such a promise problem by $\mathrm{Gap}_g\mathrm{Kt}$:

▶ **Definition 6.8.** For a function $g\colon \mathbb{N} \to \mathbb{N}$, define a promise problem $\mathrm{Gap}_g\mathrm{Kt} := (\Pi_Y, \Pi_N)$ by

$$\Pi_Y := \{\, (x,s) \in \{0,1\}^* \times \mathbb{N} \mid \mathrm{Kt}(x) \leq s \,\},$$
$$\Pi_N := \{\, (x,s) \in \{0,1\}^* \times \mathbb{N} \mid \mathrm{Kt}(x) > s + g(|x|) \,\}.$$

For this promise problem, we prove:

▶ **Theorem 1.4** (restated). *For any nondecreasing function $g(n) = \omega(\log n)$, it holds that $\mathsf{P}^{\mathrm{Gap}_g\mathrm{Kt}} \neq \mathsf{EXP}$.*

The proof is similar to a simplified proof in [1, Corollary 40] showing that resource-bounded Kolmogorov complexity K^t for a fixed exponential time $t(n) \geq 2^{n^2}$ is not EXP-hard (originally proved by Buhrman and Mayordomo [8]).

Proof. It is sufficient to prove that every unary language in $\mathsf{P}^{\mathrm{Gap}_g\mathrm{Kt}}$ can be solved in a fixed exponential time. Indeed, by the time hierarchy theorem, there exists a unary language in EXP that requires time complexity larger than the fixed exponential time, which implies that $\mathsf{P}^{\mathrm{Gap}_g\mathrm{Kt}} \neq \mathsf{EXP}$.

We first note that $\mathrm{Kt}(x) \leq |x| + O(\log |x|)$ for any $x \in \{0,1\}^*$, since every string can be described by itself in polynomial time. Let $l(n)$ be such a (nondecreasing) upper bound (*i.e.,* $l(n) = n + O(\log n)$).

Let $L \subseteq \{0\}^*$ be an arbitrary unary language in $\mathsf{P}^{\mathrm{Gap}_g\mathrm{Kt}}$, and M be a polynomial-time machine that witnesses $L \in \mathsf{P}^{\mathrm{Gap}_g\mathrm{Kt}}$.

The proof idea is as follows: We would like to simulate M on input 0^n without oracle access to $\mathrm{Gap}_g\mathrm{Kt}$ in time $2^{2n} \ll 2^{n^{O(1)}}$. To this end, we try to answer M's query q by exhaustively searching up to Kt-complexity $l(n)$. While we cannot obtain the correct value $\mathrm{Kt}(q)$ for a query q such that $\mathrm{Kt}(q) > l(n)$, we guess the value $\mathrm{Kt}(q)$ to be $l(n)$. Then, we will argue that each query q can be computed efficiently and hence $\mathrm{Kt}(q)$ is relatively small; therefore, the guessed value of Kt-complexity gives a good approximation. A formal proof follows.

We define a machine M_0 that simulates M on input 0^n (without oracle access to $\mathrm{Gap}_g\mathrm{Kt}$): On input 0^n, M_0 simulates M on the same input, and accepts if and only if M accepts. If

the machine M makes a query $(q, s) \in \{0,1\}^* \times \mathbb{N}$ to a $\mathrm{Gap}_g\mathrm{Kt}$ oracle, then we perform an exhaustive search up to Kt-complexity $l(n)$, which allows us to compute $\sigma_n(q) := \min\{\,\mathrm{Kt}(q), l(n)\,\}$. (Namely, for each $d \in \{0,1\}^*$ of length at most $l(n)$, run the universal Turing machine U on input d for time $2^{l(n)-|d|}$, which takes overall $2^{l(n)}n^{O(1)}$ time.) We answer "Yes" to the query q if and only if $\sigma_n(q) \leq s$. The machine M_0 runs in time $2^{l(n)}n^{O(1)} \leq 2^{2n}$ (*i.e.,* a fixed exponential time). Hence, it remains to prove that, for each $n \in \mathbb{N}$, there exists an oracle A that satisfies the promise of $\mathrm{Gap}_g\mathrm{Kt}$ such that $M_0(0^n) = M^A(0^n)$, which in particular implies that $M_0(0^n) = L(0^n)$.

A crucial observation here is that each query that M makes on the computation path simulated by M_0 can be described succinctly in terms of Kt-complexity: Specifically, fix an input 0^n and define the set $Q_n = \{\,(q_1, s_1), \cdots, (q_m, s_m)\,\}$ of queries that M makes on the computation path simulated by M_0, where $m = n^{O(1)}$ is the number of the queries. Then, the ith query (q_i, s_i) can be described by n and an index $i \in [m]$ in time $2^{l(n)}n^{O(1)}$. Therefore, it holds that $\mathrm{Kt}(q_i) \leq O(\log n) + \log 2^{l(n)}n^{O(1)} = l(n) + O(\log n)$. By the assumption, we have $O(\log n) \leq g(n)$ for all large n; hence, $\mathrm{Kt}(q_i) \leq l(n) + g(n)$. This means that the difference between $\mathrm{Kt}(q_i)$ and the threshold $l(n)$ up to which we performed an exhaustive search is at most $g(n)$.

Now, for each $n \in \mathbb{N}$, define an oracle A as follows: $(q, s) \in A$ if and only if $\sigma_n(q) \leq s$ for any $(q, s) \in Q_n$, and $(q, s) \in A$ if and only if $\mathrm{Kt}(q) \leq s$ for any $(q, s) \notin Q_n$. (Here, $\sigma_n(q)$ denotes $\min\{\,\mathrm{Kt}(q), l(n)\,\}$.) By this definition, it holds that $M^A(0^n) = M_0(0^n)$; thus all that remains is to show that A satisfies the promise of $\mathrm{Gap}_g\mathrm{Kt}$ (which implies that $M^A(0^n) = L(0^n)$).

Namely, for all $(q, s) \in Q_n$, we would like to claim that $(q, s) \in A$ holds if (q, s) is a YES instance of $\mathrm{Gap}_g\mathrm{Kt}$ (*i.e.,* $\mathrm{Kt}(q) \leq s$), and that $(q, s) \notin A$ holds if (q, s) is a NO instance of $\mathrm{Gap}_g\mathrm{Kt}$ (*i.e.,* $\mathrm{Kt}(q) \geq s + g(|q|)$). Note that if $\mathrm{Kt}(q) \leq l(n)$ then $\sigma_n(q) = \mathrm{Kt}(q)$; hence in this case, the claim is obviously satisfied. In what follows, we may assume that $\mathrm{Kt}(q) > l(n)$ (and thus $\sigma_n(q) = l(n)$). In particular, this implies that $n \leq |q|$: indeed, by the definition of $l(n)$, we have $\mathrm{Kt}(q) \leq l(|q|)$, which implies $l(n) < \mathrm{Kt}(q) \leq l(|q|)$; hence, $n \leq |q|$ follows. Therefore, $\mathrm{Kt}(q) \leq l(n) + g(n) \leq l(n) + g(|q|)$. Now assume that $\mathrm{Kt}(q) > s + g(|q|)$ (*i.e.,* (q, s) is a NO instance). This implies that $\sigma_n(q) = l(n) \geq \mathrm{Kt}(q) - g(|q|) > s$, and hence $(q, s) \notin A$ as desired. On the other hand, if $\mathrm{Kt}(q) \leq s$ (*i.e.,* (q, s) is an YES instance), then we have $\sigma_n(q) \leq \mathrm{Kt}(q) \leq s$, and hence $(q, s) \in A$. ◀

Next, assuming that $\mathsf{EXP} \subseteq \mathsf{P/poly}$, we translate the property of Kt-complexity into that of MCSP. However, since these two measures are just *polynomially* related, the narrow gap of Kt does not seem to be translated into a narrow gap of MCSP. Thus, we define $\mathrm{Gap}^k\mathrm{MCSP}$ as a promise problem that asks for approximating the *logarithm* of circuit complexity within a factor of k:

▶ **Definition 6.9.** For a constant $k \geq 1$, define a promise problem $\mathrm{Gap}^k\mathrm{MCSP} := (\Pi_Y, \Pi_N)$ by

$$\Pi_Y := \{\,(T, s) \in \{0,1\}^* \times \mathbb{N} \mid \log \mathrm{K}_I(T) \leq s\,\},$$
$$\Pi_N := \{\,(T, s) \in \{0,1\}^* \times \mathbb{N} \mid \log \mathrm{K}_I(T) > ks\,\}.$$

We can apply the same proof idea to $\mathrm{Gap}^k\mathrm{MCSP}$. In fact, thanks to the fact that the gap between Π_Y and Π_N is wide, we can prove a somewhat strong consequence:

▶ **Theorem 6.10.** *If* $\mathsf{EXP} \subseteq \mathsf{P/poly}$*, then for any* $\epsilon > 0$*, there exists a constant* $k \geq 1$ *such that* $\mathsf{P}^{\mathrm{Gap}^k\mathrm{MCSP}} \subseteq \mathsf{DTIME}(2^{n^\epsilon})$*. In particular,* $\mathsf{EXP} \neq \mathsf{P}^{\mathrm{Gap}^k\mathrm{MCSP}} \cap \mathsf{P/poly}$ *for some* k*.*

Proof. The proof idea is exactly the same with that of Theorem 1.4: We first simulate a $\mathsf{P}^{\mathrm{Gap}^k\mathrm{MCSP}}$ machine by answering its query T by an exhaustive search up to circuit complexity $l(n)$ for some $l(n)$. Then, since any query T can be described succinctly in terms of Kt-complexity, the circuit complexity $\mathrm{K}_I(T)$ of the query T is also relatively small by Lemma 6.2; hence, the incomplete exhaustive search gives a somewhat good approximation. While the theorem can be proved based on Lemma 6.2, we incorporate a proof of Lemma 6.2 and give an entire proof below for completeness.

Let us define an EXP-complete language $B \subseteq \{0,1\}^*$ as all the tuples $\langle Q, x, t\rangle$ such that the Turing machine Q accepts x in time t. Since $B \in \mathsf{EXP} \subseteq \mathsf{P/poly}$, there exist some constant $k_0 \in \mathbb{N}$ and some family of circuits $\{ C_m \}_{m\in\mathbb{N}}$ of size at most m^{k_0} that computes B on input length m.

Fix a small constant $\epsilon > 0$. Define $k := (k_0+1)/\epsilon$. Let $L \in \mathsf{P}^{\mathrm{Gap}^k\mathrm{MCSP}}$ and M be a polynomial-time oracle machine that witnesses $L \in \mathsf{P}^{\mathrm{Gap}^k\mathrm{MCSP}}$.

Define $l(n) := n^\epsilon$. As in the proof of Theorem 1.4, we define a machine M_0 that simulates M (without oracle access to $\mathrm{Gap}^k\mathrm{MCSP}$) as follows: M_0 takes input $x \in \{0,1\}^*$ of length n, simulates M on input x, and accepts if and only if M accepts. If M makes a query (T, s), then answer to the query by an exhaustive search up to circuit size $l(n)$. (Specifically, compute $\sigma_x(T) := \min\{ \mathrm{K}_I(T), l(n) \}$ and answer "Yes" if and only if $\sigma_x(T) \le s$.) The machine M_0 runs in time $2^{l(n)}n^{O(1)} \le 2^{n^{2\epsilon}}$ for all large n.

Fix input $x \in \{0,1\}^*$ of length n. Let $Q_x = \{ (T_1, s_1), \cdots, (T_{n^{O(1)}}, s_{n^{O(1)}}) \}$ be the set of all the queries that M makes on the computation path simulated by M_0. We claim that for each $(T_i, s_i) \in Q_x$, the circuit complexity $\mathrm{K}_I(T_i)$ is relatively small: Indeed, each truth-table T_i in Q_x can be computed in time $t(n) := 2^{n^{2\epsilon}}$, by simulating M in the same way with M_0. Let Q be the Turing machine that takes as input $x \in \{0,1\}^*$ of length n and indices $i, j \in [n^{O(1)}]$, and outputs T_{ij}. By the definition of B, it holds that $B(Q, \langle x, i, j\rangle, t(n)) = Q(x, i, j) = T_{ij}$. Also, by the definition of C_m, we have $B(Q, \langle x, i, j\rangle, t(n)) = C_m(Q, \langle x, i, j\rangle, t(n))$ for $m = |\langle Q, \langle x, i, j\rangle, t(n)\rangle|$. Note that $m = 4n + O(\log n) + \log t(n) \le 5n$ for all large n. Now let us fix $x \in \{0,1\}^n$ and $i \in [n^{O(1)}]$: namely, define $D_{x,i}(j) = C_m(Q, \langle x, i, j\rangle, t(n))$; then, the truth-table of $D_{x,i}$ coincides with T_i. Therefore,

$$\mathrm{K}_I(T_i) \le |D_{x,i}| \le |C_m| \le m^{k_0} \le (5n)^{k_0} \le n^{k\epsilon} = l(n)^k$$

for all large n. (Here, $|C_m|$ denotes the circuit size of C_m.)

Now we claim that $\sigma_x(T_i) = \min\{ \mathrm{K}_I(T_i), l(n) \}$ approximates $\mathrm{K}_I(T_i)$ for all $(T_i, s_i) \in Q_x$: specifically, we claim that $\log \sigma_x(T_i) \le \log \mathrm{K}_I(T_i) < k \log \sigma_x(T_i)$. If $\mathrm{K}_I(T_i) \le l(n)$, then $\sigma_x(T_i) = \mathrm{K}_I(T_i)$ and the claim is obvious. Now assume that $\mathrm{K}_I(T_i) > l(n)$, which implies that $\sigma_x(T_i) = l(n)$. Thus we have $\sigma_x(T_i) = l(n) < \mathrm{K}_I(T_i) < l(n)^k = \sigma_x(T_i)^k$.

From the inequalities above, for all but finitely many $x \in \{0,1\}^*$, it is easy to see that there exists an oracle A such that A satisfies the promise of $\mathrm{Gap}^k\mathrm{MCSP}$ and $M_0(x) = M^A(x) = L(x)$. ◀

As in Corollary 6.6, we obtain:

▶ **Corollary 6.11.** *If $L \le_T^p \mathrm{Gap}^k\mathrm{MCSP}$ for all $k \ge 1$, then $\mathsf{P}^L \cap \mathsf{P/poly} \ne \mathsf{EXP}$.*

Proof. The hypothesis implies that $\mathsf{P}^L \subseteq \mathsf{P}^{\mathrm{Gap}^k\mathrm{MCSP}}$ for all $k \ge 1$, and Theorem 6.10 shows $\mathsf{EXP} \not\subseteq \mathsf{P}^{\mathrm{Gap}^k\mathrm{MCSP}} \cap \mathsf{P/poly}$ for some $k \ge 1$, from which the result follows. ◀

▶ Remark.

1. As in the case of nonadaptive reductions, establishing NP-hardness of $\mathrm{Gap}^k\mathrm{MCSP}$ for all $k \ge 1$ via a polynomial-time Turing reduction implies that $\mathsf{P}^{\mathsf{NP}} \cap \mathsf{P/poly} \ne \mathsf{EXP}$.

2. One interesting consequence is that if MCSP itself is reducible to Gap^{k}MCSP for all $k \geq 1$ via a polynomial-time Turing reduction, then $\mathsf{P}^{\mathsf{MCSP}} \cap \mathsf{P/poly} \neq \mathsf{EXP}$, which we do not know how to prove. Thus, establishing such "robustness" of MCSP via a polynomial-time Turing reduction is at least as hard as separating $\mathsf{P}^{\mathsf{MCSP}} \cap \mathsf{P/poly}$ from EXP.

Finally, we observe that every language in statistical zero knowledge is reducible to Gap^{k}MCSP via a BPP-reduction. As observed in [5], hardness of statistical zero knowledge implies hardness of approximating the minimum circuit complexity of a truth-table T within a factor of $|T|^{1-\epsilon}$ for any $\epsilon \in (0,1)$. Similarly, it implies hardness of Gap^{k}MCSP for all $k \geq 1$ (*i.e.,* a problem of approximating the logarithm of the circuit complexity within an arbitrary constant factor).

▶ **Theorem 1.6** (restated). *For all $k \geq 1$, every language in statistical zero knowledge is reducible to* Gap^{k}MCSP *via a* BPP*-Turing reduction.*

Proof. Let A be an arbitrary oracle that satisfies the promise of Gap^{k}MCSP. Let $s(n) := \frac{1}{2k} \log n$. Define an oracle $B := \{\, x \in \{0,1\}^* \mid (x, s(|x|)) \notin A \,\}$. It is sufficient to show that B satisfies the hypothesis of Theorem 3.1.

First, we claim that B does not contain any string of low circuit complexity. Suppose that $x \in B$. Then we have $(x, s(|x|)) \notin A$, which implies that $(x, s(|x|))$ is not an YES instance of Gap^{k}MCSP. This means that $\log \mathrm{K}_I(x) > s(|x|)$; hence, $\mathrm{K}_I(x) > |x|^{1/2k}$.

Next, we claim that the oracle B is of polynomial density. It is sufficient to prove that $\{\, x \in \{0,1\}^* \mid \mathrm{K}_I(x) > |x|^{1/2} \,\} \subseteq B$: Indeed, suppose that $\mathrm{K}_I(x) > |x|^{1/2}$ for a string $x \in \{0,1\}^*$; then we have $\log \mathrm{K}_I(x) > ks(|x|)$, which implies that $(x, s(|x|))$ is a NO instance of Gap^{k}MCSP; hence, $x \in B$. ◀

Acknowledgements. We would like to thank anonymous referees of CCC 2016 for their helpful comments and suggestions. This work is supported in part by MEXT KAKENHI No. 24106008.

References

1 Eric Allender, Harry Buhrman, Michal Koucký, Dieter van Melkebeek, and Detlef Ronneburger. Power from random strings. *SIAM J. Comput.*, 35(6):1467–1493, 2006. doi:10.1137/050628994.

2 Eric Allender and Bireswar Das. Zero knowledge and circuit minimization. In *Proceedings of the 39th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 25–32, 2014. doi:10.1007/978-3-662-44465-8_3.

3 Eric Allender, Joshua Grochow, and Cristopher Moore. Graph isomorphism and circuit size. *Electronic Colloquium on Computational Complexity (ECCC)*, 22:162, 2015. URL: http://eccc.hpi-web.de/report/2015/162.

4 Eric Allender, Dhiraj Holden, and Valentine Kabanets. The minimum oracle circuit size problem. In *Proceedings of 32nd International Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 21–33, 2015. doi:10.4230/LIPIcs.STACS.2015.21.

5 Eric Allender, Michal Koucký, Detlef Ronneburger, and Sambuddha Roy. The pervasive reach of resource-bounded kolmogorov complexity in computational complexity theory. *J. Comput. Syst. Sci.*, 77(1):14–40, 2011. doi:10.1016/j.jcss.2010.06.004.

6 Andrej Bogdanov and Luca Trevisan. On worst-case to average-case reductions for NP problems. *SIAM J. Comput.*, 36(4):1119–1159, 2006. doi:10.1137/S0097539705446974.

7 Ravi Boppana, Johan Håstad, and Stathis Zachos. Does co-NP have short interactive proofs? *Inf. Process. Lett.*, 25(2):127–132, 1987. doi:10.1016/0020-0190(87)90232-8.

8 Harry Buhrman and Elvira Mayordomo. An excursion to the Kolmogorov random strings. *J. Comput. Syst. Sci.*, 54(3):393–399, 1997. doi:10.1006/jcss.1997.1484.

9 Lance Fortnow. The complexity of perfect zero-knowledge. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing (STOC)*, pages 204–209, 1987. doi:10.1145/28395.28418.

10 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 (STOC)*, pages 59–68, 1986. doi:10.1145/12130.12137.

11 Juris Hartmanis and Richard Stearns. On the computational complexity of algorithms. *Transactions of the American Mathematical Society*, pages 285–306, 1965.

12 Johan Håstad, Russell Impagliazzo, Leonid Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999. doi:10.1137/S0097539793244708.

13 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 (STOC)*, pages 220–229, 1997. doi:10.1145/258533.258590.

14 Valentine Kabanets and Jin-yi Cai. Circuit minimization problem. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing (STOC)*, pages 73–79, 2000. doi:10.1145/335305.335314.

15 Ker-I Ko. On the complexity of learning minimum time-bounded turing machines. *SIAM J. Comput.*, 20(5):962–986, 1991. doi:10.1137/0220059.

16 Leonid Levin. Randomness conservation inequalities; information and independence in mathematical theories. *Information and Control*, 61(1):15–37, 1984. doi:10.1016/S0019-9958(84)80060-1.

17 Cody Murray and Ryan Williams. On the (non) NP-hardness of computing circuit complexity. In *Proceedings of 30th Conference on Computational Complexity (CCC)*, pages 365–380, 2015. doi:10.4230/LIPIcs.CCC.2015.365.

New Non-Uniform Lower Bounds for Uniform Classes

Lance Fortnow[1] **and Rahul Santhanam**[*2]

1 **Georgia Institute of Technology, Atlanta, USA**
fortnow@cc.gatech.edu

2 **Department of Computer Science, University of Oxford, Oxford, United Kingdom**
rahul.santhanam@cs.ox.ac.uk

Abstract

We strengthen the nondeterministic hierarchy theorem for non-deterministic polynomial time to show that the lower bound holds against sub-linear advice. More formally, we show that for any constants d and d' such that $1 \leqslant d < d'$, and for any time-constructible bound $t = o(n^d)$, there is a language in $\mathsf{NTIME}(n^d)$ which is not in $\mathsf{NTIME}(t)/n^{1/d'}$. The best known earlier separation of Fortnow, Santhanam and Trevisan could only handle $o(\log(n))$ bits of advice in the lower bound, and was not tight with respect to the time bounds.

We generalize our hierarchy theorem to work for other syntactic complexity measures between polynomial time and polynomial space, including alternating polynomial time with any fixed number of alternations. We also use our technique to derive an *almost-everywhere* hierarchy theorem for non-deterministic classes which use a sub-linear amount of non-determinism, i.e., the lower bound holds on all but finitely many input lengths rather than just on infinitely many.

As one application of our main result, we derive a new lower bound for NP against NP-uniform non-deterministic circuits of size $O(n^k)$ for any fixed k. This result is a significant strengthening of a result of Kannan, which states that not all of NP can be solved with P-uniform circuits of size $O(n^k)$ for any fixed k. As another application, we show strong non-uniform lower bounds for the complexity class RE of languages decidable in randomized linear exponential time with one sided error.

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

Keywords and phrases Computational Complexity, Nondeterminism, Nonuniform Complexity

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.19

1 Introduction

One of the fundamental questions in complexity theory is whether resource hierarchies exist, i.e., whether having more of a resource allows us to solve more computational problems. Hierarchies are known for many fundamental resources, including deterministic time [11, 12], deterministic space [18] and non-deterministic time [6, 17, 20, 8].

Hierarchy theorems yield the only unconditional separations we know against polynomial-time classes, and thus it is of interest to investigate how strong we can make these separations.

* Work done when the author was at University of Edinburgh, Edinburgh, UK, and supported by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013)/ERC Grant Agreement no. 615075

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 19; pp. 19:1–19:14

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

Ideally, we would like the separations to work against *non-uniform* classes, not just uniform ones. The notion of *advice* allows us to interpolate between the uniform and the non-uniform settings, and then the question becomes how much advice we can handle in the lower bound when proving a hierarchy theorem.

This question is interesting for at least a couple of different reasons. First, the amount of non-uniformity in the lower bound is closely tied to the question of derandomization. If we could show that for any fixed k, there is a language in deterministic polynomial time which cannot be solved in deterministic time $O(n^k)$ with $O(n^k)$ bits of advice, we could conclude that every language in probabilistic polynomial time can be solved infinitely often in deterministic sub-exponential time, using the hardness-randomness tradeoffs of [15, 3]. A similar derandomization result for the class MA follows from the assumption that there is a language in NP which cannot be solved in non-deterministic time $O(n^k)$ with $O(n^k)$ bits of advice.

Second, from a technical point of view, hierarchy theorems are used in many of the important separation results in complexity theory [2, 7, 19]. Improved hierarchy theorems open the way to stronger versions of these results.

The traditional proofs of hierarchy theorems yield only uniform lower bounds. However, the proof of the deterministic time hierarchy theorem [11, 12] can easily be adapted to yield separations against $n - \omega(1)$ bits of advice. This adaptation exploits the closure of deterministic time under complementation.

The situation is very different for resources such as non-deterministic time which are not known to be closed under complementation. The best hierarchy theorem known for this case in terms of the advice handled by the lower bound is due to [10]. They adapt Zak's proof of the non-deterministic time hierarchy [20] to show that $\mathsf{NP} \not\subseteq \mathsf{NTIME}(n^c)/\log(n)^{1/2c}$ for any $c > 0$. Not much more can be expected of adaptations of classical proofs of the non-deterministic time hierarchy theorem [6, 17, 20]. Since such proofs consider exponentially many input lengths when diagonalizing against a single machine, they're incapable of handling advice more than $O(\log(n))$.

1.1 Our Results

Our main result is a significant improvement of the non-deterministic time hierarchy theorem in terms of the advice handled in the lower bound.

▶ **Theorem 1.1.** *Let $d \geqslant 1$ and $d' > d$ be any constants, and let t be a time-constructible time bound such that $t = o(n^d)$. Then $\mathsf{NTIME}(n^d) \not\subseteq \mathsf{NTIME}(t)/n^{1/d'}$.*

Theorem 1.1 improves on known results handling advice in two respects. First, the amount of advice in the lower bound can be as high as $n^{\Omega(1)}$, in contrast to earlier results in which it was limited to be $O(\log(n))$. Second, the hierarchy is provably tight in terms of the time bounds, while earlier results handling advice could only separate $\mathsf{NTIME}(n^d)$ from $\mathsf{NTIME}(n^c)$ with advice, where $c < d$.

The ideas of the proof of Theorem 1.1 also enable us to make progress on another direction in which hierarchy theorems can be strengthened: showing that hierarchy theorems hold almost everywhere. By this we mean that the lower bound holds on all but finitely many input lengths, rather than just on infinitely many. While it is well-known that the deterministic time hierarchy theorem can be adapted to hold almost everywhere, it is a long-standing open problem whether this adaptation can be done for the non-deterministic hierarchy theorem. It is shown in [5] that any adaptation has to be non-relativizing.

We make progress on this question by showing that almost-everywhere hierarchies do hold for a very natural sub-class of non-deterministic time: non-deterministic time with bounded non-determinism. Given functions t and g, let $\mathsf{NTIMEGUESS}(t, g)$ denote the class of languages accepted by non-deterministic machines running in time $t(n)$ and using at most $g(n)$ non-deterministic bits on any input of length n. Note that most natural NP-complete problems, such as SAT and CLIQUE, belong to $\mathsf{NTIMEGUESS}(\mathsf{poly}(n), o(n))$. We show the following.

▶ **Theorem 1.2.** *Let $d > 1$ be any constant, and let t be a time-constructible function such that $t(n) = o(n^d)$. Let $g(n) = o(n)$ be any function computable in time $O(n)$. Then* $\mathsf{NTIMEGUESS}(n^d, 2g) \not\subseteq \mathsf{i.o.NTIMEGUESS}(t, g)$.

We are able to use Theorem 1.1 to derive a new circuit lower bound for NP, improving a 30-year old result of Kannan [14].

▶ **Theorem 1.3.** *Let $k > 1$ be any constant.* NP *does not have* NP*-uniform non-deterministic circuits of size $O(n^k)$.*

We are also able to use Theorem 1.1 to derive improved non-uniform lower bounds for the complexity class RE of problems solvable in randomized linear exponential time with one-sided error. Previously only a separation against a logarithmic amount of advice was known [5].

▶ **Theorem 1.4.** *For any constant c,* $\mathsf{RE} \not\subseteq \mathsf{RTIME}(n^c)/n^{1/2c}$.

Finally, we consider the question of whether Theorem 1.1 can be extended to complexity measures other than NTIME. We show that for a wide variety of complexity measures, including all the alternating time classes with a bounded number of alternations, the analogue of Theorem 1.1 holds. Since the statements of these results are somewhat technical, we refer the reader to Section 7.

1.2 Techniques

We now attempt to give some intuition for the ideas in our proofs.

Recall that we are attempting to give hierarchies for non-deterministic time where the upper bound is uniform, but the lower bound allows as large an amount of non-uniformity as possible. Tradtional proofs of uniform non-deterministic time hierarchy theorems [6, 17, 20] use the delayed diagonalization technique. We illustrate this technique through Zak's proof, which is arguably the simplest. Suppose we wish to define a non-deterministic machine M running in time n^d which diagonalizes against some non-deterministic machine M_i running in time $t = o(n^d)$. Rather than diagonalizing against M_i on some fixed input x depending on i as in the proof of the deterministic time hierarchy theorem [11, 12], we diagonalize against M_i on some interval I_i of input lengths, meaning that we are guaranteed M differs from M_i on some input of length in I_i. The interval I_i is of the form $[n_i, 2^{n_i^d}]$ for some n_i depending on i, though we overload the notation I_i to also represent the set of strings whose length is in that interval. The diagonalization proceeds via a "copying" mechanism. On an input x in I_i of length less than $2^{n_i^d}$, M on x simply simulates M_i on $x0$, accepting iff M_i accepts. On an input of the form $x0^{2^{n_i^d}-n_i}$, where $|x| = n_i$, M determines $M_i(x)$ by brute force search, accepting iff M_i rejects. By assumption on t and assuming n_i is large enough, M can be defined to run in time n^d on all inputs in I_i.

Assume, for the purpose of contradiction, that M and M_i define the same language. Then M and M_i agree on all inputs with lengths in I_i, which by the copying mechanism of M,

implies that $M_i(x)$ agrees with $M_i(x0^j)$ for each x of length n_i and each $j \in [0, 2^{n_i^d} - n_i]$. But then M cannot agree with M_i on $x0^{2^{n_i^d}-n_i}$, as M on that input does the opposite of what M_i does on x. Note that we cannot guarantee that M differs from M_i on any specific input, merely that it differs from M_i on some input in I_i. Also note that the interval I_i is exponentially long. Intuitively, M bides its time for exponentially many input lengths, until it has enough resources to do the opposite of what M_i does on x.

With an appropriate choice of intervals I_i, the above argument yields a uniform hierarchy theorem. It was adapted by Fortnow, Santhanam and Trevisan [9] to show a hierarchy with advice, but the advice which the adaptation can handle is very low: $o(\log(n))$. To handle advice, M needs to simulate M_i with advice in the copying phase. The advice used is extracted from x in a deterministic way, so that considering all possible strings x of a certain length enables us to diagonalize against all possible advice strings of a smaller length. However, the fact that Zak's argument uses exponentially many input lengths hurts us in terms of the amount of advice we can handle. First, using a naive copying argument requires an exponential amount of information (advice bits for all input lengths in the interval) to be encoded into the starting input x, which is impossible. This is dealt with in [9] by only using sub-logarithmically input lengths in an exponentially long interval I_i, namely input lengths of the form $n_i^{c^k}$, where c is a large enough constant and k varies, and "jumping" from one input length m to a polynomially larger one m^c during the copy phase. The cost paid for the way this issue is dealt with in [9] is that the time bounds in the hierarchy theorem are polynomially separated rather than just being asymptotically separated as in the proof of the uniform non-deterministic time hierarchy. There is also a second issue, which is that for Zak's form of delayed diagonalization to work, advice for the final input length in the interval must be encoded into x. This constrains the advice that can be handled in this argument to sub-logarithmic, as the final input length in the interval is exponentially larger than x.

This second issue is a bottleneck for all delayed diagonalization arguments using exponentially long intervals, which includes all the traditional arguments [6, 17, 20]. Recently, Fortnow and Santhanam [8] gave a new proof of the non-deterministic time hierarchy theorem, which unlike previous proofs, critically uses the definition of non-deterministic time using polynomial-time verifiability. This new argument has the benefit that it uses only a polynomially long interval, and is a natural starting point for an attempt to handle more advice in the non-deterministic time hierarchy.

Intuitively, rather than "copying along a line" as in Zak's argument, the Fortnow-Santhanam proof "copies down a tree". Suppose we wish to define a non-deterministic machine M running in time n^d which diagonalizes against some non-deterministic machine M_i running in time $t = o(n^d)$. We again define some interval I_i of input lengths for achieving this, but now $I_i = [n_i, n_i + n_i^d]$ is only polynomially long. For any input $y \in I$ of length less than $n_i + n_i^d$, M copies the behaviour of M_i on two different inputs of length one larger, by accepting iff both $M_i(x0)$ and $M_i(x1)$ accept. On input of the form $xw, |x| = n_i, |w| = n_i^d$, M simulates M_i on x with witness w and does the opposite. Thus this diagonalization phase actually use the non-deterministic nature of M_i, rather than simply doing brute force search. It is again easy to see that if I_i is chosen appropriately, M can be made to run in time n^d.

Now assume, for the purpose of contradiction, that M agrees with M_i on all inputs in M_i. If M_i accepts on x, then by the copying behaviour of M, M_i accepts on all inputs in the interval I. But this implies that for all candidate witnesses w of size n_i^d, M_i rejects on x with witness w, which is a contradiction, as M_i would then reject on x itself. The case where M_i rejects on x is argued similarly.

By using only a polynomially long interval, the argument above, which we term witness-based diagonalization, gives hope for handling a sub-polynomial amount of advice in the

lower bound. However, there are again obstacles to adapting the argument to advice. Even if the argument uses a polynomially long interval, it still uses all input lengths within that interval. A naive adaptation of the argument would require advice for all these input lengths to be encoded into x, which would be impossible as the number of input lengths is larger than x.

We could try using jumps again, so that fewer input lengths within the interval are used. However, it is unclear how to do this with witness-based diagonalization, as every jump only contributes to one bit in the witness, and therefore with a small number of jumps, we are unable to build a witness which we can use in the diagonalization process at the last input length in the interval.

We solve the problem by hybridizing between delayed diagonalization and witness-based diagonalization. The idea is that witness-based diagonalization can be "simulated" within a single input length, namely the last input length in the interval. However, in order to perform this simulation, we need to copy from the first input length in the interval to the last one. This can be done using jumps again, but how we use jumps critically affects the parameters in the final hierarchy results. The fewer the jumps used, the more advice we can handle, but the larger the gap between the time upper bound and the time lower bound. We need to choose the jump mechanism appropriately to get an optimal tradeoff between the quality of the ensuing hierarchy theorem in terms of time bounds and the quality of the ensuing hierarchy theorem in terms of advice. This gets somewhat technical, but we are able to prove Theorem 1.1 using these ideas.

The proof of Theorem 1.1 still uses a polynomially long interval for diagonalization. Suppose we wish to prove an almost-everywhere hierarchy for non-deterministic time, i.e., a hierarchy theorem where the lower bound holds for almost all input lengths rather than for infinitely many lengths[1]. It is known [5] that this cannot be done in a relativizing way. We show in this paper that an almost-everywhere hierarchy *can* be obtained for a natural subclass of non-deterministic time, namely non-deterministic time with sub-linear witnesses. The key observation is that when the amount of non-determinism is sub-linear, a variant of the witness-based diagonalization argument can be carried out within a single input length, meaning that we can diagonalize against any fixed machine on *any* large enough input length. This yields an almost-everywhere hierarchy.

The proof of Theorem 1.3 is substantially different. It uses an indirect diagonalization technique due to [16], where the presumed existence of a simulation of a class C with weakly uniform circuits of fixed polynomial size is used multiple times to derive a simulation of C in a small amount of time with sub-linear advice, as long as the uniformity condition is in some sense stronger than the class C. We require a variant of this argument which uses a census technique, and then an application of Theorem 1.1 completes the proof.

The proof of Theorem 1.4 uses a win-win analysis. Either Satisfiability has efficient randomized algorithms with sub-polynomial advice, or it does not. We show that the statement of Theorem 1.4 holds in either case, with the argument in the first case depending on Theorem 1.1. Note that the proof technique of Theorem 1.1 is not directly applicable to classes such as randomized polynomial time for which computable enumerations of the languages in the class are not known, however we are still able to use indirect arguments to derive interesting lower bounds for such classes.

For the extensions to other complexity measures, we abstract out the properties required

[1] Note that this notion of almost-everywhere separations is different from the related notion considered by [1], who give a negative relativization result

of the complexity measure using the notion of *leaf languages* introduced by Bovet, Crescenzi and Silvestri [4]. This enables us to establish analogues of Theorem 1.1 for the levels of the polynomial-time hierarchy, as well as counting classes such as $\mathsf{C_{=}P}$.

2 Preliminaries

2.1 Complexity Classes, Promise Problems and Advice

We assume a basic familiarity with complexity classes. The Complexity Zoo (which can be found at `http://qwiki.caltech.edu/wiki/ComplexityZoo`) is an excellent resource for basic definitions and statements of results. We use the multitape Turing machine model and assume all our functions are time constructible as described in a standard textbook [13].

We require some classes defined by simultaneous resource bounds. Let $t : \mathbb{N} \to \mathbb{N}$ be a time bound, and $g : \mathbb{N} \to \mathbb{N}$ be a bound on the amount of non-determinism used. The complexity class $\mathsf{NTIMEGUESS}(t, g)$ is the class of all languages L for which there is a non-deterministic machine M deciding L which runs in time $O(t(n))$ and uses at most $g(n)$ guess bits on any input of length n.

Given a complexity class C, coC is the class of languages L such that $\bar{L} \in \mathsf{C}$. Given a function $s : \mathbb{N} \to \mathbb{N}$, $\mathsf{SIZE}(s)$ is the class of Boolean functions $f = \{f_n\}$ such that for each n, f_n has Boolean circuits of size $O(s(n))$. Given a language L and an integer n, $L_n = L \cap \{0,1\}^n$. Given a class C, $\mathsf{i.o.C}$ is the class of languages L for which there is a language $L' \in \mathsf{C}$ such that $L_n = L'_n$ for infinitely many length n.

In order to deal with promise classes in a general way, we take as fundamental the notion of a complexity measure. A complexity measure CTIME is a mapping which assigns to each pair (M, x), where M is a time-bounded machine (here a time function $t_M(x)$ is implicit) and x an input, one of three values "0" (accept), "1" (reject) and "?" (failure of CTIME promise). We distinguish between *syntactic* and *semantic* complexity measures. Syntactic measures have as their range $\{0, 1\}$ while semantic measures may map some machine-input pairs to "?". The complexity measures DTIME and NTIME are syntactic (each halting deterministic or non-deterministic machine either accepts or rejects on each input), while complexity measures such as BPTIME and MATIME are semantic (a probabilistic machine may accept on an input with probability $1/2$, thus failing the bounded-error promise). For syntactic measures, any halting machine defines a language, while for semantic measures, only a subset of halting machines define languages.

Let $t : \mathbb{N} \to \mathbb{N}$ be a time function, and $a : \mathbb{N} \to \mathbb{N}$ be an advice function. A language L is in $\mathsf{CTIME}(t)/a$ if there is a machine M halting in time $t(\cdot)$ taking an auxiliary *advice* string of length $a(\cdot)$ such that for each n, there is some advice string $b_n, |b_n| = a(n)$ such that M fulfils the CTIME promise for each input x with advice string b_n and accepts x iff $x \in L$.

We will need standard notions of uniformity for circuits. The *direct connection language* for a sequence of circuits $C = \{C_n\}$, where C_n is on n input bits, is the language L_C consisting of all tuples of the form $\langle 1^n, g, h, r\rangle$, where g and h are indices of gates, r is the *type* of g (AND/OR/NOT/INPUT, and in case of INPUT, which of the n input bits g is, with an additional bit to specify whether g is the designated output gate), and h is a gate feeding in to g in case the type r is not INPUT. Other encodings of the direct connection language are of course possible, but our results are insensitive to the details of the encoding.

Given a class $\mathcal{C}$ of languages and a function $s : \mathbb{N} \to \mathbb{N}$, a language L is said to have $\mathcal{C}$-uniform circuits of size $s(n)$ if there is a size-$s(n)$ circuit family $\{C_n\}$ such that its *direct connection language* is computable in $\mathcal{C}$. By a *description of a circuit* C_n, we mean the list of tuples in L_C corresponding to gates in C_n.

The complexity measure RTIME corresponds to randomized time with one-sided error, defined by probabilistic machines which, for each input, either accept with probability at least $1/2$ or reject with probability 1. The class $\mathsf{RE} = \mathsf{RTIME}(2^{O(n)})$. We also use $\mathsf{E} = \mathsf{DTIME}(2^{O(n)})$.

3 Hierarchies for Non-deterministic Time against Sublinear Advice

In this section, we prove the following general theorem, and then show how it implies Theorem 1.1.

As described in the Introduction section, the proof involves a hybrid of delayed diagonalization and witness-based diagonalization. We think of the diagonalization as proceeding in two phases: the jump phase where copying occurs, and the witness-gathering phase where the witness is built and witness-based diagonalization is performed.

We need some preliminary notation. Let $f : \mathbb{N} \to \mathbb{N}$ be a function such that $f(n)$ is computable in $O(\mathsf{polylog}(n))$ time and $f(n) > n$ for all n. We will use f to parameterize the jumps in the diagonalization. Given a time function t_1, for any n, let $g(n)$ be the minimum i such that $f^{(i)}(n) \geqslant n + 2t_1(n) + 2$ where $f^{(i)}$ is f applied to itself i times. Note that for each $n, g(n)$ exists, using the monotonicity of f. For a string w of length r, we define $Enc(w)$ to be the $2r$-bit string whose even bits are all 0, and whose i'th odd bit is the i'th bit of w, for each $i \in [r]$.

▶ **Theorem 3.1.** *Let t_1 and t_2 be increasing time-constructible functions, with $t_1, t_2 = \Omega(n)$. Let $f, g : \mathbb{N} \to \mathbb{N}$ be functions as defined above, and let $a : \mathbb{N} \to \mathbb{N}$ be an advice function such that $a(n)$ is computable in time $O(\mathsf{polylog}(n))$. Suppose $n = \sum_{l=0}^{g(n)} a(f^{(l)}(n)) + \omega(1)$, and $t_1(f(m)) + g(m)\mathsf{polylog}(m) = o(t_2(m))$. Then $\mathsf{NTIME}(t_2) \not\subseteq \mathsf{NTIME}(t_1)/a$*

Proof. Define a non-deterministic machine M as follows. On input x of length m, M first calculates $t_2(m)$. It then tries to decompose $x = 1^i01^j0z110^k$, where $i, j > 0$, $k \geqslant 0$, $z \in \{0,1\}^*$. Note that such a decomposition is unique if it exists. If M succeeds in finding such a decomposition, it sets $n = i+j+|z|+4$, and checks if $m = f^l(n)$ for some $0 \leqslant l \leqslant g(n)$, and if $|z| \geqslant \sum_{l=0}^{g(n)} a(f^{(l)}(n))$. This check can be done in time at most $g(n)\mathsf{polylog}(n)$ and hence time at most $g(m)\mathsf{polylog}(m)$, by assumption on f and g. If this check doesn't succeed, M rejects. If it succeeds, there are two cases: $l < g(n)$ and $l = g(n)$. In the first case, M decomposes $z = z_0z_1 \dots z_{l+1}z'$, where for each $i, 0 \leqslant i \leqslant l+1, |z_i| = a(f^{(i)}(n))$ and $z' \in \{0,1\}^*$. Note that by assumption on n and a, such a decomposition can be performed for n large enough – if it cannot be performed, M halts and rejects. M simulates M_i on $x0^{f(m)-m}$ with advice z_{l+1}, accepting iff M_i accepts. In the second case, where $l = g(n)$, M decomposes $z = z_0z_1 \dots z_lz'$, where for each $i, 0 \leqslant i \leqslant l, |z_i| = a(f^{(i)}(n))$ and $z' \in \{0,1\}^*$. Note that by assumption on n and a, such a decomposition can be performed for n large enough – if it cannot be performed, M halts and rejects. It also calculates $q = k - 2t_1(n) - 2$. Note that q is non-negative by the assumptions on f and g. M simulates M_i on $1^i01^j0z11Enc(0^{t_1(n)}1)0^q$ with advice z_l, accepting iff M_i accepts. Throughout M maintains an internal clock, and if it detects that it has been running for more than $t_2(m)$ steps after the calculation of $t_2(m)$, it halts and rejects.

The operation of M above corresponds to the jump phase.

Now suppose M does not succeed in finding a decomposition as above. It then tries to decompose $x = 1^i01^j0z11Enc(0^s1w)0^q$, where $i, j > 0$, $s, q \geqslant 0$, $z, w \in \{0,1\}^*$ and moreover, setting $n = i + j + |z| + 4$, the conditions that $m = f^{(g(n))}(n)$ and $|z| \geqslant \sum_{l=0}^{g(n)} a(f^{(l)}(n))$ are satisfied. Note that such a decomposition is unique if it exists. If this decomposition attempt

fails, M halts and rejects. If it succeeds, M decomposes $z = z_0 z_1 \dots z_l z'$, where for each $i, 0 \leqslant i \leqslant l, |z_i| = a(f^{(i)}(n))$ and $z' \in \{0,1\}^*$. Note that by assumption on n and a, such a decomposition can be performed for n large enough – if it cannot be performed, M halts and rejects. Now there are two cases: $s > 0$ and $s = 0$. In the first case, M simulates M_i on $1^i01^j0z11Enc(0^{s-1}1w0)0^q$ with advice z_l and $1^i01^j0z11Enc(0^{s-1}1w1)0^q$ with advice z_l, accepting iff both computations accept. In the second case, M simulates M_i on 1^i01^j0z11 with non-deterministic sequence w and advice z_0, rejecting iff M_i accepts. Throughout M maintains an internal clock, and if it detects that it has been running for more than $t_2(m)$ steps after the calculation of $t_2(m)$, it halts and rejects.

The operation of M above corresponds to the witness-gathering phase.

By definition of M, it halts in time $O(t_2(m))$. Moreover, using the various assumptions on computability of f, a, t_1, t_2, all the checks and calculations of M, as well as the final simulation step, can be completed in time $O(t_2(m))$ for m large enough.

We now proceed to show that $L(M) \notin \mathsf{NTIME}(t_1(m))/a(m)$. Suppose, to the contrary, that M_i is a non-deterministic advice taking machine accepting $L(M)$ using $a(m)$ bits of advice. We derive a contradiction.

Choose j and n large enough so that all the checks, calculations and simulation of M can be completed in time $O(t_2(m))$ for any m such that there is an input of length m which can be successfully decomposed with the corresponding n and j, and so that $n > \sum_{l=0}^{g(n)} a(f^{(l)}(n))$. Let $z_0, z_1, \dots z_{g(n)}$ be the correct advice strings for M_i at lengths $n, f(n) \dots f^{g(n)}(n)$, and let $z = z_0 z_1 \dots z_{g(n)}$. Consider the input $x = 1^i01^j0z11$. By assumption, M on x agrees with M_i on x with advice z_0 (since $|x| = n$). By the behaviour of M in the jump phase, we have that M on $x0^{f^i(n)-n}$ agrees with M_i on $x0^{f^i(n)-n}$ with advice z_i, for each $i \in [0, g(n)]$. By the behaviour of M in the witness-gathering phase, we have that M accepts $x0^{f^i(n)-n}$ iff M accepts $xEnc(0^s1w)0^q$ for each $s, 0 \leqslant s \leqslant t_1(n)$, w of length $t_1(n)-s$ and $q = m-n-2t_1(n)-2$ iff M_i accepts $xEnc(0^s1w)0^q$ with advice $z_{g(n)}$ for each $s, 0 \leqslant s \leqslant t_1(n)$, w of length $t_1(n)-s$ and $q = m-n-2t_1(n)-2$. But for each w of length $t_n(n)$, again by the behaviour of M in the witness-gathering phase, M accepts $xEnc(1w)0^q$, $q = m-n-2t_1(n)-2$ iff M_i rejects x with witness w and advice z_0. This happens iff M_i rejects x with advice z_0, which is a contradiction to the assumption that M on x agrees with M_i on x with advice z_0. ◀

We now show how to derive Theorem 1.1 from the more general Theorem 3.1 above. This allows us to get the "best of both worlds" for non-deterministic time hierarchies with advice: time bounds only asymptotically separated, and advice in the lower bound which is $n^{\Omega(1)}$.

Proof. Proof ofTheorem 1.1 Apply Theorem 3.1 with $t_2 = n^d, t_1 = t, f(n) = 2n, a(n) = n^{1/d'}$. In this case, $g(n) = O(\log(n))$, and it can be checked easily that the conditions on f, g, a in terms of t_1, t_2, n all hold. The theorem follows. ◀

4 An Almost-everywhere Hierarchy Theorem

Ideally, we would like to prove almost-everywhere hierarchy theorems, i.e., show that reducing the amount of time available makes languages harder to compute on all but finitely many input lengths. Almost-everywhere hierarchy theorems are known for classes closed under complementation such as deterministic time and deterministic space, but not for non-deterministic time. It is shown in [5] that there is an oracle relative to which $\mathsf{NEXP} \subseteq \mathsf{i.o.NP}$, therefore non-standard techniques would be required even to show an almost-everywhere separation of NEXP from NP.

We consider non-deterministic classes with sub-linear non-determinism, i.e., the non-deterministic machine is allowed to use only $o(n)$ non-deterministic bits. These classes contain most commonly studied problems in NP including $SAT, CLIQUE, VC$ etc. when the input is encoded in the standard way. Thus showing an almost-everywhere hierarchy for such classes is of interest.

The following theorem immediately implies Theorem 1.2.

▶ **Theorem 4.1.** *Let $g(n) = o(n)$ be any sub-linear function computable in time $O(n)$. Let t_1 and t_2 be time-constructible functions such that $n \leqslant t_1 = o(t_2)$. Then* $\mathsf{NTIMEGUESS}(t_2, 2g(n)) \not\subseteq \mathsf{i.o.NTIMEGUESS}(t_1, g(n))$.

Proof. Define a non-deterministic machine M as follows. On input x of length n, M first tries to decompose $x = 1^i01^k0z$, where $i, k \geqslant 1$. If x cannot be decomposed in this manner, or if it can but $|z| > g(n)$, M immediately rejects. If $|z| = g(n)$, M runs the non-deterministic Turing machine M_i on $1^i01^{n-i-2}0$ for at most $t_2(n)$ steps, using z as the sequence of guess bits for the simulation of the machine. If the machine M_i does not halt within time $t_2(n)$, or if it uses more than $g(n)$ guess bits, M rejects. Otherwise, it does the opposite of M_i, accepting if M_i rejects and rejecting otherwise.

If $|z| < g(n)$, M runs M_i on $x_1 = 1^i01^{k-1}00z$ and $x_2 = 1^i01^{k-1}01z$, accepting iff both simulations halt and accept within time $t_2(n)$, and each uses at most $g(n)$ guess bits.

M runs in time $O(t_2(n))$ and uses at most $2g(n)$ guess bits on any input of length n. We show that $L(M) \notin \mathsf{i.o.NTIMEGUESS}(t_1(n), g(n))$.

Suppose, to the contrary, that $L(M) \in \mathsf{i.o.NTIMEGUESS}(t_1(n), g(n))$, and let M_i be a non-deterministic machine running in time $ct_1(n)$ for some constant c, and with $g(n)$ guess bits, such that $L(M_i)$ coincides with $L(M)$ on infinitely many input lengths. Let I be an infinite set of input lengths such that $L(M_i)$ coincides with $L(M)$ on each input length in I. Choose $n \in I$ large enough such that M can complete its simulations of M_i on all inputs of length n of the form 1^i0y for some y. That such an n exists follows from the facts that $n \leqslant t_1(n) = o(t_2(n))$.

By the assumption that M agrees with M_i on length n, we have that M_i accepts $1^i01^{n-i-2}0$ iff M accepts $1^i01^{n-i-2}0$ iff M_i accepts $1^i01^{n-i-3}00$ and $1^i01^{n-i-3}10$... Continuing inductively, we have that M_i accepts $1^i01^{n-i-2}0$ iff M accepts all strings of the form $1^i01^{n-g(n)-i-2}0z$ iff M_i does not accept on $1^i01^{n-i-2}0$ for any guess sequence z of length $g(n)$. But then we have that M_i accepts $1^i01^{n-i-2}0$ iff M_i does not accept $1^i01^{n-i-2}0$, which is a contradiction. ◀

By combining the ideas in the proof of Theorem 4.1 with the ideas of the proof of Theorem 3.1, we get the following almost-everywhere hierarchy against advice. We omit the proof because it contains no new ideas beyond those in the proofs of Theorem 4.1 and Theorem 3.1.

▶ **Theorem 4.2.** *Let $a : \mathbb{N} \to \mathbb{N}$ be an advice function and $g : \mathbb{N} \to \mathbb{N}$ a guess function, both computable in time $O(n)$, such that $a(n) + g(n) = n - \omega(1)$. Then for any time-constructible functions t_1 and t_2 such that $n \leqslant t_1 = o(t_2)$,* $\mathsf{NTIMEGUESS}(t_2, 2g) \not\subseteq \mathsf{i.o.NTIMEGUESS}(t_1, g)/a$.

5 A Lower Bound against Weakly Uniform Circuits

While it is a major open problem to show that NP does not have linear size circuits, one could hope to show lower bounds when there is some uniformity condition on the circuits. A result of this form was shown by [14].

▶ **Theorem 5.1** ([14]). *For every k,* NP *does not have* P*-uniform circuits of size $O(n^k)$.*

We strengthen this lower bound in two ways. First, we allow the circuits to be NP-uniform rather than P-uniform. Second, we allow the circuits to be non-deterministic rather than deterministic. The following is a re-statement of Theorem 1.3.

▶ **Theorem 5.2.** *For every $k > 1$,* NP *does not have* NP*-uniform non-deterministic circuits of size $O(n^k)$.*

Proof. Assume NP has NP-uniform non-deterministic circuits of size $O(n^k)$. Let $L \in \mathsf{NP}$ be arbitrary. We will show that L can be simulated in non-deterministic time n^{2k+2} with $n^{1/(4k)}$ bits of advice, which will yield a contradiction to Theorem 3.1 when $t_2 = n^{4k}$ and $t_1 = n^{2k+2}$.

By assumption, L has non-deterministic circuits of size $O(n^k)$, so there is a non-deterministic circuit family $\{C_n\}$ for L of size at most $c \cdot n^k$ for some constant c. Furthermore, by NP-uniformity, the direct connection language L_{dc} for $\{C_n\}$ (see Section 2 for the definition) is in NP. We consider a "succinct" version L_{succ} of the language L_{dc}, defined as follows. Letting $Bin(n)$ be the binary representation of n, define

$$L_{succ} = \{\langle Bin(n)01^{\lceil n^{1/(5k^2)} \rceil}, g, h, r\rangle \mid \langle 1^n, g, h, r\rangle \in L_{dc}\}.$$

Intuitively, L_{succ} is an "unpadded" version of L_{dc}.

Observe that $L_{succ} \in \mathsf{NP}$. Given an input y for L_{succ}, our non-deterministic polynomial-time algorithm first checks if y can be parsed as a "valid" tuple $\langle z, g, h, r\rangle$, where $z = Bin(n)01^{\lceil n^{1/(5k^2)} \rceil}$ for some positive integer n, g and h are valid gate indices between 1 and $c \cdot n^k$, and r is a valid gate type. If this check fails, *reject*. Otherwise, the algorithm runs the non-deterministic polynomial-time machine deciding L_{dc} on $\langle 1^n, g, h, r\rangle$, and *accepts* if and only if this machine accepts. Note that this algorithm for L_{succ} runs in time polynomial in $|y|$, since we only simulate the machine for L_{dc} when $n^{1/(5k^2)} \leqslant |y| \leqslant n$ and the machine for L_{dc} runs in time polynomial in n.

Now we apply the assumption that NP has NP-uniform circuits of size $O(n^k)$ for a second time. Since $L_{succ} \in \mathsf{NP}$, there is a non-deterministic circuit family $\{D_m\}$ of $O(m^k)$ size for L_{succ}. Given an integer n, let $m(n)$ be the least integer such the size of the tuple $\langle Bin(n)01^{\lceil n^{1/(5k^2)} \rceil}, g, h, r\rangle$ is at most $m(n)$ for any valid gate indices g and h for C_n and any valid gate type r. Using a standard encoding of tuples, we can assume, for large enough n, that $m(n) \leqslant n^{1/(4.5k^2)}$, since g, h, r can all be encoded with $O(\log n)$ bits each.

We now describe a simulation of L in time $O(n^{2k+2})$ with $n^{1/(4k)}$ bits of advice. Let M be an advice-taking machine which operates as follows. On input x of length n, M receives an advice string of length $O(n^{1/4k})$. It interprets this advice as consisting of two parts: the description of a non-deterministic circuit D_m for the language L_{succ} on inputs of length $m(n) \leqslant n^{1/(4.5k^2)}$, and an $O(\log(n))$ bit string representing the census value, i.e., the number of inputs in L_{succ} of that length. For every possible pair of gate indices g and h of C_n and every possible gate type r, M simulates the circuit D_m on $\langle Bin(n)01^{\lceil n^{1/(5k^2)} \rceil}, g, h, r\rangle$ to decide whether gate h is an input to gate g and whether the type of gate g is r. Each such simulation can be done in time $O(n^{1/2k})$, as the size of D_m is $O(n^{1/4k})$. There are at most $O(n^{2k+1})$ such simulations that M performs, since there are at most that many relevant triples $\langle g, h, r\rangle$. Note that since the circuit D_m is non-deterministic, M cannot know for sure the answer to a given simulation. Instead, it performs all the simulations and then checks that the number of YES answers is equal to the census value encoded in the advice string. In such a case, it knows that the answers to all simulations are correct; otherwise, it rejects.

In the case where answers to all simulations are correct, M has a full description of the non-deterministic circuit C_n. It simulates C_n on x, and accepts if and only if $C_n(x)$ outputs 1. This simulation can be done in time $O(n^{2k})$ since the circuit C_n is of size $O(n^k)$. The total time taken by M is $O(n^{2k+2})$, and M uses $O(n^{1/4k})$ bits of advice. By our assumptions on C_n and D_m, the simulation is correct. Thus $L \in \mathsf{NTIME}(n^{2k+2})/O(n^{1/4k})$.

However, as $L \in \mathsf{NP}$ was chosen to be *arbitrary*, we have $\mathsf{NP} \subseteq \mathsf{NTIME}(n^{2k+2})/O(n^{1/4k})$, which for $k > 1$ contradicts Theorem 3.1. ◀

The proof of Theorem 5.2 above is closely related to a proof of Santhanam and Williams [16], who generalized Theorem 5.1 in a different direction, by showing that for any k, P does not have P-uniform circuits of size $O(n^k)$. The additional ingredients in the proof of Theorem 5.2 in comparison to the previous paper are the use of Theorem 3.1 and the use of a census technique to deal with NP-uniformity.

6 A Lower Bound for Randomized Time against Advice

In this section, we use Theorem 3.1 to prove a strong lower bound for randomized exponential time against sub-polynomial advice. Though the proof technique of Theorem 3.1 exploits the syntactic nature of the complexity measure NTIME by using an enumeration of non-deterministic machines, we show that the result is useful even for studying semantic classes such as randomized exponential time.

Buhrman, Fortnow and Santhanam [5] prove various lower bounds for semantic exponential-time classes against polynomial time with advice. Though they obtain strong lower bounds for MATIME and BPTIME, their result for RTIME is fairly weak – their proof techniques only yield that $\mathsf{RE} \not\subseteq \mathsf{RP}/O(\log(n))$. Using the new hierarchy for non-deterministic time against advice, we obtain a significant strengthening of their result. The theorem below is a re-statement of Theorem 1.4 in the introduction.

▶ **Theorem 6.1.** *For any constant c,* $\mathsf{RE} \not\subseteq \mathsf{RTIME}(n^c)/n^{1/2c}$.

Proof. Let SAT denote the satisfiability problem for CNF formulae. SAT is NP-complete by the Cook-Levin theorem, and the brute-force search algorithm for SAT implies SAT in E, using a standard encoding where the number of variables in the formula is at most the length of the encoding of the formula.

We consider two cases. Either SAT is in $\mathsf{BPP}/n^{1/2c}$, or it is not. In the first case, using downward self-reducibility of SAT to eliminate the advice, we have that SAT is in $\mathsf{BPTIME}(2^{n^{1/2c}}\mathsf{poly}(n))$. Again using downward self-reducibility to find witnesses for satisfiable SAT instances and thereby eliminating error in the case where the formula is unsatisfiable, we get that SAT is in $\mathsf{RTIME}(2^{n^{1/2c}}\mathsf{poly}(n))$. Using the fact that every language in $\mathsf{NTIME}(n^{1.5c})$ has a polynomial-time reduction to SAT where the output length of the reduction is $O(n^{1.5c}\mathsf{polylog}(n))$, we have that $\mathsf{NTIME}(n^{1.5c}) \subseteq \mathsf{RE}$. In this case, it follows from Theorem 3.1 that $\mathsf{RE} \not\subseteq \mathsf{NTIME}(n^c)/n^{1/c}$, and hence that $\mathsf{RE} \not\subseteq \mathsf{RTIME}(n^c)/n^{1/c}$.

In the other case, we have that SAT is not in $\mathsf{BPP}/n^{1/2c}$, and hence that SAT is not in $\mathsf{RP}/n^{1/2c}$. Since SAT is in E, we have that $\mathsf{E} \not\subseteq \mathsf{RP}/n^{1/2c}$, and since $\mathsf{E} \subseteq \mathsf{RE}$, we derive that $\mathsf{RE} \not\subseteq \mathsf{RP}/n^{1/2c}$ in this case.

Thus, in either case, we have that $\mathsf{RE} \not\subseteq \mathsf{RTIME}(n^c)/n^{1/2c}$. ◀

Theorem 6.1 is close to the best we can hope to show without settling long-standing open questions in computational complexity. If the advice in the lower bound could be strengthened from $n^{1/2c}$ to n^c, we would have that $\mathsf{NEXP} \not\subseteq \mathsf{SIZE}(poly)$, which would be a breakthrough circuit lower bound result.

7 Generalizing to Other Syntactic Classes

In this section we show how to generalize Theorem 3.1. We first show how to generalize the robustly-often time hierarchy of [8], and then sketch how to use the ideas of the proof to generalize Theorem 3.1.

First, we define robustly-often simulations.

Let S be a subset of positive integers. S is robust if for each positive integer k, there is a positive integer $m \geqslant 2$ such that $n \in S$ for all $m \leqslant n \leqslant m^k$.

Let L be a language, C a complexity class, and S a subset of the positive integers. We say $L \in \mathsf{C}$ on S if there is a language $L' \in \mathsf{C}$ such that $L_n = L'_n$ for any $n \in S$.

Given a language L and complexity class C, $L \in \mathsf{r.o.C}$ if there is a robust S such that $L \in C$ on S. In such a case, we say that there is a robustly-often (r.o.) simulation of L in C. We extend this notion to complexity classes in the obvious way – given complexity classes B and C, $\mathsf{B} \subseteq \mathsf{r.o.C}$ if there for each language $L \in \mathsf{B}$, $L \in \mathsf{r.o.C}$.

Now we describe a general framework in which we can show robustly-often hierarchies and hierarchies with sub-linear advice.

Let N be a nondeterministic polynomial-time Turing machine where on input x of length n, $N(x)$ has $2^{p(n)}$ computation paths indexed by strings $z \in \{0,1\}^{p(n)}$. We can also think of z as representing an integer between 1 and $2^{p(n)}$ in a standard way.

Define $\mathsf{OUTPUT}(N, x)$ to be the string w of length $2^{p(n)}$ such that zth bit of w is 1 if $N(x)$ accepts on the path indexed by z and 0 otherwise.

Let $A \subseteq \Sigma^*$. We define the class $\mathsf{LEAF}(A)$ as the class of languages L such that for some nondeterministic polynomial-time Turing machine N, $x \in L$ if and only if $\mathsf{OUTPUT}(N, x) \in A$. For example if A is the set of strings with at least one 1 then $\mathsf{LEAF}(A) = \mathsf{NP}$. We can also define $\mathsf{LEAFTIME}(A, t)$ where we restrict N to run in time $O(t)$.

We say a class $\mathcal{C}$ is closed under linear-time monotone 2-query transductions if for every language $L' \in \mathcal{C}$ and every deterministic linear-time oracle machine O making at most 2 queries to its oracle and outputting a monotone function of the answers to the queries, $L(O^{L'}) \in \mathcal{C}$. This definition might seem involved, but in fact any natural complexity arising from a leaf language satisfies this property, e.g., the levels of the polynomial-time hierarchy.

We can generalize the robustly-often hierarchy for non-deterministic time [8] as follows.

▶ **Theorem 7.1.** *Suppose A is computable by a family of* DLOGTIME*-time uniform* NC^1 *circuits. If t_1 and t_2 are functions such that t_1 is time-constructible and*

- $t_1(n+1) = o(t_2(n))$,
- $n \leqslant t_1(n) \leqslant n^c$ *for some constant c, and*
- $\mathsf{LEAFTIME}(A, t_1(n))$ *is uniformly closed under linear-time monotone 2-query transductions,*

then $\mathsf{LEAFTIME}(A, t_2(n)) \not\subseteq \mathsf{r.o.LEAFTIME}(A, t_1(n))$.

Proof. Without loss of generality assume A is computed by fan-in 2 circuits where every path has length $d \log n$ and negations are only on the inputs.

Let $M_1, M_2, \ldots$ be an enumeration of multitape nondeterministic machines that run in time $t_1(n)$. For an input x of length n, $\mathsf{OUTPUT}(M_i, x)$ will have length $2^{t_1(n)}$ and the circuit C used to determine if $\mathsf{OUTPUT}(M_i, x)$ is in A will have depth $dt_1(n)$. C has $2^{t_1(n))}$ inputs which we express as y_z for $z \in \{0,1\}^{t_1(n)}$.

Define a nondeterministic Turing machine M that on input $1^i 0 1^m 0 w$ does as follows:

- If $|w| < dt_1(i+m+2)$ consider the gate g that is reached in C by following the path w. The type of the gate g can be determined in linear time, using the fact that A is computed by $\mathsf{DLOGTIME}$-uniform log-depth circuits.

- If g is an OR gate then accept if both $M_i(1^i01^m0w0)$ and $M_i(1^i01^m0w1)$ accepts.
- If g is an AND gate then accept if either $M_i(1^i01^m0w0)$ or $M_i(1^i01^m0w1)$ accepts.

- If $|w| = dt_1(i+m+2)$ consider the input variable y_z.
  - If the variable is not negated then accept if $M_i(1^i01^m0)$ rejects on the path specified by z.
  - If the variable is negated then accept if $M_i(1^i01^m0)$ accepts on the path specified by z.

Since we can universally simulate $t(n)$-time nondeterministic multitape Turing machines on an $O(t(n))$-time 2-tape nondeterministic Turing machine and $\mathsf{LEAFTIME}(A, t_1)$ is closed under linear-time monotone 2-query transductions, $L(M) \in \mathsf{LEAFTIME}(A, O(t_1(n+1))) \subseteq \mathsf{LEAFTIME}(A, t_2(n))$.

Suppose $\mathsf{LEAFTIME}(A, t_2(n)) \subseteq \mathsf{r.o.LEAFTIME}(A, t_1(n))$. Pick a C such that $dt_1(n) \ll n^c$ for all n large enough. By the definition of r.o. there is some n_0 and a language $L \in \mathsf{LEAFTIME}(t_1(n))$ such that $L(M) = L$ on all inputs of length between n_0 and n_0^C. Fix i such that $L(x) = A(\mathsf{OUTPUT}(M_i, x))$ with $n_0 \leqslant |x| \leqslant n_0^C$. Then $z \in L(M_i) \Leftrightarrow z \in L(M)$ for all $z = 1^i01^{n_0}0w$ for $w \leqslant t_1(i+n_0+2)$.

By induction on the gates $M_i(1^i01^{n_0}0)$ accepts iff $C(\mathsf{OUTPUT}(M_i, 1^i01^{n_0}0))$ outputs false and thus iff $\mathsf{OUTPUT}(M_i, 1^i01^{n_0}0)$ is not in A. This contradicts our assumption that $L(1^i01^{n_0}0)) = A(\mathsf{OUTPUT}(M_i, 1^i01^{n_0}0))$. ◀

▶ **Corollary 7.2.** *Let $t_1, t_2 : \mathbb{N} \to \mathbb{N}$ be functions such that t_1 is time-constructible and $t_1(n+1) = o(t_2(n))$. For every integer $k \geqslant 1$, $\Sigma_k - \mathsf{TIME}(t_2) \not\subseteq \mathsf{r.o.}\Sigma_k - \mathsf{TIME}(t_1)$, and $\Pi_k - \mathsf{TIME}(t_2) \not\subseteq \mathsf{r.o.}\Pi_k - \mathsf{TIME}(t_1)$.*

We can combine the proofs of Theorem 7.1 and Theorem 3.1 to generalize Theorem 1.1 for LEAFTIME.

▶ **Theorem 7.3.** *Suppose A' is computable by* DLOGTIME*-time uniform NC^1 circuits. Let $d \geqslant 1$ and $e > d$ be arbitrary constants. If t_1 is a time-constructible function such that*

- *$t_1(n) = o(n^d)$,*
- *$\mathsf{LEAFTIME}(A', t_1(n))$ is closed under linear time monotone 2-query transductions,*

then $\mathsf{LEAFTIME}(A', n^d) \not\subseteq \mathsf{LEAFTIME}(A', t_1(n))/n^{1/e}$.

Proof Sketch. We show how to modify the proof of Theorem 3.1 for LEAFTIME.

The jump phase will remain exactly the same. In the witness gathering phase, we need to change things a little. The string w obtained from a successful decomposition of the input x in the witness-gathering phase will now correspond to a path in the circuit C accepting the leaf language which determines the answer of M_i on x. Again, we will assume without loss of generality that C is a balanced logarithmic-depth circuit, where all input-output paths are of the same length. There are two cases: w encodes a maximum-length path in C, or it does not. In the former case, let g be the gate that is reached following the path described by w. If g is an OR gate, then M simulates M_i on $1^i01^j0z11Enc(0^{s-1}1w0)0^q$ with advice z_l and $1^i01^j0z11Enc(0^{s-1}1w1)0^q$ with advice z_l, accepting iff both computations accept. If g is an AND gate, M simulates M_i on $1^i01^j0z11Enc(0^{s-1}1w0)0^q$ with advice z_l and $1^i01^j0z11Enc(0^{s-1}1w1)0^q$ with advice z_l, accepting iff either computation accepts. If w encodes a maximum-length path, let y_z be the variable pointed to by w, where z is a witness for M on x. If y_z is un-negated, M does the opposite of M_i on x using witness z with advice z_0, and if y_z is negated, M does the same as M_i on x using witness z with advice z_0.

We now get a contradiction following an argument similar to the proof of Theorem 3.1. ◀

▶ **Corollary 7.4.** *For any reals $1 \leqslant r < s$ and every integer $k \geqslant 1$, $\Sigma_k - \mathsf{TIME}(n^s) \not\subseteq \Sigma_k - \mathsf{TIME}(n^r)/n^{1/s}$ and $\Pi_k - \mathsf{TIME}(n^s) \not\subseteq \Pi_k - \mathsf{TIME}(n^r)/n^{1/s}$.*

Acknowledgments. We thank an anonymous referee for pointing out an error in a previous version of this paper.

References

1. Eric Allender, Richard Beigel, Ulrich Hertrampf, and Steven Homer. Almost-everywhere complexity hierarchies for nondeterministic time. *Theoretical Computer Science*, 115(2):225–241, 19 July 1993.
2. Eric Allender and Vivek Gore. A uniform circuit lower bound for the permanent. *SIAM Journal on Computing*, 23(5):1026–1049, 1994.
3. László Babai, Lance Fortnow, Noam Nisan, and Avi Wigderson. BPP has subexponential time simulations unless EXPTIME has publishable proofs. *Computational Complexity*, 3(4):307–318, 1993.
4. D. Bovet, P. Crescenzi, and R. Silvestri. A uniform approach to define complexity classes. *Theoretical Computer Science*, 104:263–283, 1992.
5. Harry Buhrman, Lance Fortnow, and Rahul Santhanam. Unconditional lower bounds against advice. In *Proc. of 36th Int'l Colloquium on Automata, Languages and Programming*, pages 195–209, 2009.
6. Stephen Cook. A hierarchy for nondeterministic time complexity. In *Proc. of the 4th Annual ACM Symp. on Theory of Computing*, pages 187–192, Denver, Colorado, 1–3 May 1972.
7. Lance Fortnow, Richard Lipton, Dieter van Melkebeek, and Anastasios Viglas. Time-space lower bounds for satisfiability. *Journal of the ACM*, 52(6):833–865, 2005.
8. Lance Fortnow and Rahul Santhanam. Robust simulations and significant separations. In *Proc. of the 38th Int'l Colloquium on Automata, Languages and Programming*, pages 569–580, 2011.
9. Lance Fortnow, Rahul Santhanam, and Luca Trevisan. Promise hierarchies. *Electronic Colloquium on Computational Complexity (ECCC)*, 11(98), 2004.
10. Lance Fortnow, Rahul Santhanam, and Luca Trevisan. Hierarchies for semantic classes. In *Proc. of the 37th Annual ACM Symp. on Theory of Computing*, 2005.
11. Juris Hartmanis and Richard Stearns. On the computational complexity of algorithms. *Trans. Amer. Math. Soc. (AMS)*, 117:285–306, 1965.
12. Frederick Hennie and Richard Stearns. Two-tape simulation of multitape Turing machines. *Journal of the ACM*, 13(4):533–546, October 1966.
13. J. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Language, and Computation.* Addison–Wesley, Reading, MA, 1979.
14. Ravi Kannan. Circuit-size lower bounds and non-reducibility to sparse sets. *Information and Control*, 55(1):40–56, 1982.
15. Noam Nisan and Avi Wigderson. Hardness vs randomness. *Journal of Computer and System Sciences*, 49(2):149–167, 1994.
16. Rahul Santhanam and Ryan Williams. On medium-uniformity and circuit lower bounds. In *Proc. of the 28th Annual IEEE Conf. on Computational Complexity*, pages 15–23, 2013.
17. Joel Seiferas, Michael Fischer, and Albert Meyer. Separating nondeterministic time complexity classes. *Journal of the ACM*, 25(1):146–167, January 1978.
18. Richard Stearns, Juris Hartmanis, and Philip Lewis. Hierarchies of memory limited computations. In *Proc. of the 6th Annual Symp. on Switching Circuit Theory and Logical Design*, pages 179–190. IEEE, 1965.
19. Ryan Williams. Non-uniform ACC circuit lower bounds. In *Proc. of 26th Annual IEEE Conf. on Computational Complexity*, pages 115–125, 2011.
20. Stanislav Žák. A Turing machine time hierarchy. *Theoretical Computer Science*, 26(3):327–333, October 1983.

New Characterizations in Turnstile Streams with Applications

Yuqing Ai[1], Wei Hu[2], Yi Li[3], and David P. Woodruff[4]

1 **Institute for Theoretical Computer Science (ITCS), Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing, China**
ayq12@mails.tsinghua.edu.cn
2 **Institute for Theoretical Computer Science (ITCS), Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing, China**
huw12@mails.tsinghua.edu.cn
3 **Facebook, Inc., Menlo Park, USA**
leeyi@umich.edu
4 **IBM Research – Almaden, San Jose, USA**
dpwoodru@us.ibm.com

Abstract

Recently, [Li, Nguyen, Woodruff, STOC'2014] showed any 1-pass constant probability streaming algorithm for computing a relation f on a vector $x \in \{-m, -(m-1), \ldots, m\}^n$ presented in the turnstile data stream model can be implemented by maintaining a linear sketch $A \cdot x \bmod q$, where A is an $r \times n$ integer matrix and $q = (q_1, \ldots, q_r)$ is a vector of positive integers. The space complexity of maintaining $A \cdot x \bmod q$, not including the random bits used for sampling A and q, matches the space of the optimal algorithm[1].

We give multiple strengthenings of this reduction, together with new applications. In particular, we show how to remove the following shortcomings of their reduction:

1. *The Box Constraint.* Their reduction applies only to algorithms that must be correct even if $\|x\|_\infty = \max_{i \in [n]} |x_i|$ is allowed to be much larger than m at intermediate points in the stream, provided that $x \in \{-m, -(m-1), \ldots, m\}^n$ at the end of the stream. We give a condition under which the optimal algorithm is a linear sketch even if it works only when promised that $x \in \{-m, -(m-1), \ldots, m\}^n$ at all points in the stream. Using this, we show the first super-constant $\Omega(\log m)$ bits lower bound for the problem of maintaining a counter up to an additive ϵm error in a turnstile stream, where ϵ is any constant in $(0, \frac{1}{2})$. Previous lower bounds are based on communication complexity and are only for relative error approximation; interestingly, we do not know how to prove our result using communication complexity. More generally, we show the first super-constant $\Omega(\log m)$ lower bound for additive approximation of ℓ_p-norms; this bound is tight for $1 \le p \le 2$.
2. *Negative Coordinates.* Their reduction allows x_i to be negative while processing the stream. We show an equivalence between 1-pass algorithms and linear sketches $A \cdot x \bmod q$ in dynamic graph streams, or more generally, the strict turnstile model, in which for all $i \in [n]$, $x_i \ge 0$ at all points in the stream. Combined with [Assadi, Khanna, Li, Yaroslavtsev, SODA'2016], this resolves the 1-pass space complexity of approximating the maximum matching in a dynamic graph stream, answering a question in that work.
3. 1-*Pass Restriction.* Their reduction only applies to 1-pass data stream algorithms in the turnstile model, while there exist algorithms for heavy hitters and for low rank approximation which provably do better with multiple passes. We extend the reduction to algorithms which make any number of passes, showing the optimal algorithm is to choose a new linear sketch at the beginning of each pass, based on the output of previous passes.

[1] Note the [LNW14] reduction does not lose a $\log m$ factor in space as they claim if it maintains $A \cdot x \bmod q$ rather than $A \cdot x$ over the integers.

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 20; pp. 20:1–20:22

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

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases communication complexity, data streams, dynamic graph streams, norm estimation

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.20

1 Introduction

In the turnstile streaming model [6, 10], there is an underlying n-dimensional vector x which is initialized to $\vec{0}$. The data stream consists of updates of the form $x \leftarrow x + e_i$ or $x \leftarrow x - e_i$, where e_i is the i-th standard unit vector in $\mathbb{R}^n$. The goal of a streaming algorithm is to make one or more passes over the stream and use limited memory to approximate a function of x with high probability.

1.1 Linear Sketches and Simultaneous Communication Complexity

All known algorithms for problems in the turnstile model have a similar form: they first choose a (possibly random) integer matrix A, then maintain the "linear sketch" $A \cdot x$ in the stream, and finally output a function of $A \cdot x$. Li et al. [9] showed that any 1-pass constant probability streaming algorithm for approximating an arbitrary function f of x in the turnstile model can be reduced to an algorithm which, before the stream begins, samples a matrix A uniformly from $O(n \log m)$ hardwired integer matrices, then maintains the linear sketch $A \cdot x \bmod q$, where $q = (q_1, \ldots, q_r)$ is a vector of positive integers and r is the number of rows of A. Furthermore, the logarithm of the number of all possibilities for $A \cdot x \bmod q$, as x ranges over $\{-m, -(m-1), \ldots, m\}^n$, plus the number of random bits for sampling A, is larger than the space used by the original algorithm for approximating f by at most an additive $O(\log n + \log\log m)$ bits. Here the extra $O(\log n + \log\log m)$ bits are used only for sampling A and q. We refer to this as the LNW reduction[2].

The LNW reduction is non-uniform, i.e., the space complexity does not count the number of bits to store the $O(n \log m)$ hardwired possible sketching matrices A, nor does it count the space to compute the output given $A \cdot x$. The space counts only the space of storing $A \cdot x$. Thus, the LNW reduction is mostly useful in proving *lower bounds*, which only become stronger by not counting some parts in the space complexity. A widely used technique for proving lower bounds on the space of streaming algorithms is communication complexity [15]. One can get a 1-pass space lower bound for any streaming algorithm $\mathcal{A}$ by constructing a communication problem in which the players create data streams based on their inputs and run $\mathcal{A}$ on these streams sequentially. At the end of each stream, the current player passes the memory contents of $\mathcal{A}$ to the next player, and the next player continues with the received intermediate state. If $\mathcal{A}$ outputs the correct answer for the communication problem with constant probability at the end of the stream of the last player, then the space of $\mathcal{A}$ is at least the one-way communication complexity of the communication problem divided by (the number of players $-$ 1).

The LNW reduction makes it possible to instead consider the *simultaneous communication model*. Compared to one-way communication, the simultaneous communication model is a

[2] In [9] the linear sketch of the form $A \cdot x \bmod q$ is further reduced to the form $A \cdot x$, which leads to a multiplicative $O(\log m)$ factor loss in the space complexity, assuming $m = \text{poly}(n)$; we do not consider that further reduction in this paper.

more restrictive model in which each player can only send a single message to an additional player called the referee, who receives no input in the communication problem. The referee then announces its output. By reducing an algorithm in an arbitrary form to a linear sketch and exploiting the linearity of matrix multiplication, the LNW reduction shows that to obtain lower bounds in the turnstile model, it suffices to consider the simultaneous communication model. This technique was applied in the original paper [9] and followup work for estimating frequency moments [13].

1.2 Shortcomings of the LNW Reduction

The LNW reduction has several drawbacks, which we now describe.

The Box Constraint. First, the reduction can only be performed under the assumption that the algorithm works as long as the underlying vector x belongs to $\{-m, -(m-1), \ldots, m\}^n$ *at the end* of the stream, while in certain settings a more natural requirement may be that x belongs to $\{-m, -(m-1), \ldots, m\}^n$ *at all intermediate points* of the stream. We refer to the restriction that the algorithm must be correct (with constant probability) provided that $x \in \{-m, -(m-1), \ldots, m\}^n$ at the end of the stream, even if $\|x\|_\infty > m$ at an intermediate point, as *the box constraint*. It is possible that there are more space-efficient algorithms, not based on linear sketches, which abort if $\|x\|_\infty$ ever becomes larger than m. Due to this reason, the lower bounds obtained via simultaneous communication complexity only apply to the class of streaming algorithms assuming the box constraint.

Negative Coordinates. The second drawback is that the reduction works only in the turnstile model which allows negative frequencies, and does not work in the *strict turnstile model* in which the underlying vector always has no negative entries. For graph problems, a multi-graph with n vertices is defined as a stream in which each update corresponds to the addition or the deletion of an edge between two vertices. The multiplicity of every edge is naturally required to be always non-negative, so the strict turnstile model is standard for graph problems. The input for graphs in this model is called a *dynamic graph stream*. Similar to the turnstile model, linear sketching is the only existing technique for designing streaming algorithms in dynamic graph streams. It is unknown whether there is an equivalence between linear sketches and single-pass algorithms in the strict turnstile model.

1-Pass Restriction. Another shortcoming of the LNW reduction is that it only applies to 1-pass data stream algorithms in the turnstile model, while there exist algorithms for heavy hitters and for low rank approximation which provably do better with multiple passes. It is unknown if there exists a similar characterization for multi-pass algorithms.

1.3 Our Contributions

We make significant progress on removing the above shortcomings of the LNW reduction.

The Box Constraint. We give a condition under which the box constraint on the algorithm can be removed. Under this condition, we show that the streaming algorithm can be reduced to a linear sketch if it is correct with constant probability for streams whose underlying vector x always belongs to $\{-m, -(m-1), \ldots, m\}^n$ at any point in the stream. In other words, we do not require algorithms to be correct when $\|x\|_\infty > m$ at intermediate points in the stream. Consequently, when our condition is satisfied, the lower bounds obtained via

simultaneous communication complexity are stronger since the box constraint on algorithms is removed. Our condition for removing the box constraint is that the algorithm has space complexity at most $O((\log m)/n)$; so it is most useful when $m \gg n$ or $n = O(1)$. Note that while this does not apply to a number of data stream problems, it does apply to some very fundamental ones, described below, such as maintaining a counter in a stream, for which $n = 1$. We also show our condition that the space be $O((\log m)/n)$ bits is tight in the sense that for larger space algorithms, the LNW reduction fails unless one allows $\|x\|_\infty > m$ at intermediate points. That is, we give an example of an algorithm with $\Omega((\log m)/n)$ bits of space for which if one applies the LNW reduction, to argue correctness one needs $\|x\|_\infty > m$.

Negative Coordinates. We show in the *strict turnstile model*, there is a reduction from a general 1-pass algorithm to a linear sketch, that is, the optimal algorithm is a linear sketch even if promised that $x_i \geq 0$ at all points in the stream. Here we assume the space complexity of the algorithm depends only on n, even if the underlying vector is allowed to have very large entries at intermediate points in the stream. This assumption is suitable for graph problems for which the desired lower bounds are usually in terms of n [2]. Note that for graph and multi-graph problems with edge weight multiplicity bounded by $\text{poly}(n)$, such a condition does not affect known upper bounds. Indeed, known algorithms are linear sketches, for which each coordinate can be maintained modulo $\text{poly}(n)$.

1-Pass Restriction. We extend the reduction to algorithms which make any number of passes, showing the optimal algorithm is to choose a new linear sketch at the beginning of each pass, based on the output of previous passes. We note that in [7], significantly better bounds for finding ℓ_2-heavy hitters were found using multiple passes, while in [3] the 2-round protocol in the arbitrary partition model there can be implemented as a 2-pass streaming algorithm with better space than possible of any 1-pass algorithm [14].

1.4 Applications

Norm Approximation and Maintaining a Counter

A fundamental problem in the turnstile streaming model is norm approximation [1], in which the goal is to output an approximation of the ℓ_p-norm $\|x\|_p = (\sum_{i=1}^n |x_i|^p)^{\frac{1}{p}}$, for given $p > 0$.[3] In particular, we are interested in proving space lower bounds for any 1-pass algorithm that outputs *additive error* approximation of the ℓ_p-norm: for $x \in \{-m, -(m-1), \ldots, m\}^n$, the algorithm outputs a number in $\left[\|x\|_p - \epsilon n^{1/p} m, \|x\|_p + \epsilon n^{1/p} m\right]$ with high probability. Since we have $\|x\|_p \leq n^{1/p} m$ for all $x \in \{-m, -(m-1), \ldots, m\}^n$, a $(1 \pm \epsilon)$-relative error approximation implies an $(\pm \epsilon n^{1/p} m)$-additive error approximation; however, an additive error approximation is much weaker. In some applications, relative error is too restrictive and one may only be interested in the value of a norm if it is sufficiently large. However, all previous lower bounds, e.g., [8], only apply if the norm is allowed to be very small, that is, they do not apply to additive error approximation.

We obtain the first super-constant $\Omega(\log m)$ lower bound for approximating $\|x\|_p$ up to an *additive* $\epsilon n^{1/p} m$ error in the turnstile model, without any assumptions such as the box constraint, where ϵ is any constant in $(0, 1/2)$. Our lower bound of $\Omega(\log m)$ bits is optimal for the important case of $p \in [1, 2]$, which includes the Manhattan and Euclidean norms.

[3] For $0 < p < 1$, $\|x\|_p$ is not a norm, though it is still a well-defined function.

Indeed, for $p \in [1, 2]$ one can obtain a relative error approximation using $O(\log m)$ bits of space [1, 5].

Previous lower bound techniques are based on two-player communication complexity in which the players, Alice and Bob, hold inputs $x \in \{-m, -(m-1), \ldots, m\}^n$ and $y \in \{-m, -(m-1), \ldots, m\}^n$ respectively, and should output an approximation to $\|x - y\|_p$. But for an additive error approximation, it suffices for Alice to send the most significant $O(1)$ bits of x to Bob, and thus only an $\Omega(1)$ lower bound can be proved via communication complexity.

For the special case of $n = 1$, the data stream is composed of $+1$'s and -1's and the underlying vector x is an integer, i.e., a "counter". When x is promised to stay in $\{-m, -(m-1), \ldots, m\}$, we are interested in the space complexity of maintaining $|x|$ up to an additive ϵm error, for constant $\epsilon \in (0, 1/2)$. Surprisingly, the space complexity of this problem in the turnstile model with additive error was previously unknown. There is an obvious $O(\log m)$ upper bound for this problem because the algorithm can just maintain x. The question is whether this upper bound is tight. By removing the box constraint of the LNW reduction, we give the first tight $\Omega(\log m)$ bits lower bound for this fundamental problem. As a simple corollary, we show that outputting the most significant bit of $|x|$ in a turnstile stream requires $\Omega(\log m)$ bits of space. Here we let $|x|$ take $(\lfloor \log m \rfloor + 1)$ bits so its most significant bit can be 0. Note that this is in sharp contrast to maintaining the least significant $O(1)$ bits, which can be done with $O(1)$ bits of space. Indeed, if one is interested in the C least significant bits, it suffices to maintain a counter modulo 2^C.

Matching Problems

Matching problems are among the most studied graph problems in the streaming model. In a recent work [2], Assadi et al. give a 1-pass algorithm using $\tilde{O}(n^{2-3\epsilon})$ bits of space to recover an n^ϵ-approximate maximum matching in dynamic graph streams. They also show a lower bound of $n^{2-3\epsilon-o(1)}$ bits for any linear sketch that approximates the maximum matching to within a factor of $O(n^\epsilon)$. Their bounds are essentially tight for linear sketches, but it remains to see whether they are also tight for general 1-pass algorithms.

Our result for non-negative streams implies that the upper and lower bounds in [2] for approximating maximum matching are tight not only for linear sketches, but for all 1-pass algorithms. Thus the space complexity of approximating maximum matching in dynamic graph streams is resolved.

2 Preliminaries

We present some notations and definitions in this section.

Data Streams in the Turnstile Model. Let e_i be the i-th standard unit vector in $\mathbb{R}^n$. In the turnstile streaming model, the input $x \in \mathbb{Z}^n$ is represented as a data stream $\sigma = (\sigma_1, \sigma_2, \ldots)$ in which each element σ_i belongs to $\Sigma = \{e_1, \ldots, e_n, -e_1, \ldots, -e_n\}$ and $\sum_i \sigma_i = x$.

The frequency of a stream σ is denoted by $\operatorname{freq} \sigma = \sum_i \sigma_i$. For two streams σ and τ, let $\sigma \circ \tau$ be the stream obtained by concatenating τ to the end of σ. The inverse stream of σ, denoted by σ^{-1}, is defined inductively by $e_i^{-1} = -e_i$, $(-e_i)^{-1} = e_i$ and $(\sigma \circ \tau)^{-1} = \tau^{-1} \circ \sigma^{-1}$.

Let $\Lambda_m = \{\sigma \mid \|\operatorname{freq} \sigma\|_\infty \le m\}$ and $\Gamma_m = \{\sigma \mid \text{for any prefix } \sigma' \text{ of } \sigma, \|\operatorname{freq} \sigma'\|_\infty \le m\}$; the former is the set of input streams without the removal of the box constraint, and the latter the set of input streams when the box constraint is removed.

The Strict Turnstile Model. In the strict turnstile model, there is an additional requirement that the underlying vector should not have negative coordinate at any intermediate point. In other words, this model allows input streams in $\Lambda_m^* = \{\sigma \mid \|\operatorname{freq}\sigma\|_\infty \leq m$, and for any prefix σ' of σ, $\sigma' \geq \vec{0}\}$.

Stream Automata. A stream automaton $\mathcal{A}$ is a Turing machine that uses two tapes, a unidirectional read-only input tape and a bidirectional work tape. The input tape contains the input stream σ. After processing its input, the automaton writes an output, denoted by $\phi_{\mathcal{A}}(\sigma)$, on the work-tape. A configuration of $\mathcal{A}$ is determined by its state of the finite control, head position and contents on the work tape. We often use the word "state" to mean a configuration. The computation of $\mathcal{A}$ can be described by a transition function $\oplus : C \times \Sigma \to C$, where C is the set of all possible configurations. For a configuration $c \in C$ and a stream σ, we also denote by $c \oplus \sigma$ the configuration after processing σ on c. The set of configurations of $\mathcal{A}$ that are achievable by some input stream $\sigma \in \Gamma_m$ is denoted by $C(\mathcal{A}, m)$. The space of $\mathcal{A}$ with stream parameter m is then defined to be $S(\mathcal{A}, m) = \log|C(\mathcal{A}, m)|$.

A problem P is characterized by a family of binary relations $P_n \subseteq \mathbb{Z}^{p(n)} \times \mathbb{Z}^n$, where $n \geq 1$ and $p(n)$ is the dimension of the output. We say an automaton $\mathcal{A}$ solves a problem P (with domain size n) on a distribution Π if $(\phi_{\mathcal{A}}(\sigma), \operatorname{freq}\sigma) \in P_n$ with probability $1 - \delta$, where the probability is over $\sigma \sim \Pi$ (and where δ is a small positive constant specified when needed).

Path-Reversible Automata and Path-Independent Automata. An automaton is said to be path-reversible if for any configuration c and any input stream σ, $c \oplus (\sigma \circ \sigma^{-1}) = c$. An automaton is said to be path-independent if for any configuration c and any input stream σ, $c \oplus \sigma$ depends only on freq σ and c.

Transition Graph. The transition graph of an automaton $\mathcal{A}$ is a directed graph $G_{\mathcal{A}} = (V, E)$, where the vertex set V is the set of configurations of $\mathcal{A}$, and the arcs in E describe the transition function of $\mathcal{A}$: there is an arc a from vertex c_1 to c_2 if and only if there is an update $u \in \Sigma$ such that $c_1 \oplus u = c_2$, and we denote this update u by f_a. Note that every vertex in V has $2n$ outgoing arcs, each of which corresponds to a possible update in Σ.

Zero-Frequency Path and Zero-Frequency Graph. In a transition graph $G_{\mathcal{A}} = (V, E)$, a path p of length k from v_1 to v_{k+1} is a sequence of k arcs $(v_1, v_2), (v_2, v_3), \ldots, (v_k, v_{k+1})$. Let $f_p = \sum_{i=1}^{k} f_{(v_i, v_{i+1})}$ be the frequency of path p, which is the frequency of the stream along p. A path p is called a zero-frequency path if $f_p = \vec{0}$. The zero-frequency graph $G'_{\mathcal{A}} = (V', E')$ based on $G_{\mathcal{A}} = (V, E)$ is a directed graph with $V' = V$ and $E' = \{(v_1, v_2) \mid$ there exists a zero-frequency path from v_1 to v_2 in $G_{\mathcal{A}}\}$.

Randomized Stream Automata. A randomized stream automaton is a deterministic automaton with one additional tape for the random bits. The random bit string R is initialized on the random bit tape before any input token is read; then the random bit string is used in a bidirectional read-only manner. The rest of the execution proceeds as in a deterministic automaton. A randomized automaton $\mathcal{A}$ is said to be path-independent (reversible) if, for each possible randomness R, the deterministic instance $\mathcal{A}_R$ is path-independent (reversible). The space of a randomized automaton $\mathcal{A}$ with stream parameter m is defined as $S(\mathcal{A}, m) = \max_R(|R| + S(\mathcal{A}_R, m))$.

Multi-Pass Stream Automata. A p-pass ($p \geq 2$) deterministic automaton $\mathcal{A}$ consists of automata of p layers (passes), in which (i) the 1-st pass automaton $\mathcal{A}^1$ which contains a starting state, when reading an input stream σ and arriving at state s, outputs a second-pass deterministic automaton $\mathcal{A}^2_s$; (ii) for $2 \leq q \leq p-1$, a q-th pass automaton $\mathcal{A}^q$, when reading input σ and arriving at a state s, outputs a $(q+1)$-th pass deterministic automaton $\mathcal{A}^{q+1}_s$; (iii) a p-th pass automaton $\mathcal{A}^p$, when reading input σ and arriving at a state s, outputs a final answer for the input σ. When the context is clear, we also use σ to mean the terminating state of the automaton when reading stream σ, e.g., $\mathcal{A}^2_\sigma$ is the same as $\mathcal{A}^2_{o \oplus \sigma}$, where o is the initial state of $\mathcal{A}^2$.

For an input stream σ, a sequence of automata will be generated, $\mathcal{A}^1, \mathcal{A}^2_{s_1}, \ldots, \mathcal{A}^p_{s_{p-1}}$, where s_i $(1 \leq i \leq p-1)$ is the terminating state of $\mathcal{A}^i_{s_{i-1}}$ (where $\mathcal{A}^1_{s_0} = \mathcal{A}^1$) on reading σ. A p-pass deterministic automaton $\mathcal{A}$ solves a problem P on input stream σ if the output of $\mathcal{A}^p_{s_{p-1}}$ on σ is an acceptable answer for σ. We say that the answer is *acceptable* for σ in this case. The space complexity $S(\mathcal{A}, m)$ is defined as $S(\mathcal{A}, m) = \max_{\sigma:\|\text{ freq}(\sigma)\|_\infty \leq m}\{S(\mathcal{A}^1, m) + S(\mathcal{A}^2_{s_1}, m) + \ldots + S(\mathcal{A}^p_{s_{p-1}}, m)\}$.

A p-pass automaton is said to be path-independent if all of its constituent automata are path-independent. A p-pass randomized automaton is defined similarly as a 1-pass randomized automaton.

3 Removing the Box Constraint and Application to Additive Error Norm Approximation

In this section, we give a condition under which the box constraint can be removed. As an application, we obtain an $\Omega(\log m)$ bit lower bound for the additive error ℓ_p-norm $\|x\|_p (p > 0)$ estimation in the turnstile streaming model.

First of all, we remark that the LNW reduction in [9] can be simplified. The LNW reduction consists of three main steps: (i) reduction from a general automaton to a path-reversible automaton; (ii) reduction from a path-reversible automaton to a path-independent automaton; (iii) reduction from a path-independent automaton to a linear sketch $A \cdot x$.[4] We notice that step (ii) is not necessary, since the automaton obtained after step (i), which was shown to be path-reversible in [9], is already path-independent. The proof of this fact in given in Appendix A.

3.1 Removing the Box Constraint in the Turnstile Model

In the LNW reduction, given a problem P, if an algorithm $\mathcal{A}$ solves P on Λ_m, then it is reduced to a linear sketch that solves P on Λ_m. (Recall that Λ_m is the set of all streams σ with $\|\text{freq } \sigma\|_\infty \leq m$.) Our goal is to remove the box constraint, i.e., to apply the reduction to algorithms that solve P on Γ_m, which is the set of streams σ such that $\|\text{freq } \sigma'\|_\infty \leq m$ for any prefix of σ' of σ. We will show that if $\mathcal{A}$ uses space $S(\mathcal{A}, m) \leq c \cdot \frac{\log m}{n}$ for some fixed constant $c > 0$ and solves P on Γ_m, then it can be reduced to a linear sketch that solves P on $\Gamma_{m/2}$. The reason why the LNW reduction requires $\mathcal{A}$ to solve the problem on Λ_m instead of Γ_m comes from the step of reducing a general automaton to a path-independent automaton. Given a certain deterministic instance of the general automaton $\mathcal{A}$, the transition graph $G_\mathcal{A} = (V, E)$ and the zero-frequency graph $G'_\mathcal{A} = (V', E')$ are built. The states of

[4] Note that a path-independent automaton is equivalent to maintaining a linear sketch $A \cdot x \bmod q$ [4, 9]. The only goal of step (iii) is to remove the "mod q".

a corresponding deterministic instance of the new automaton $\mathcal{B}$ are defined to be all the *terminal* strongly connected components[5] of $G'_{\mathcal{A}}$. The transition function of $\mathcal{B}$ is then defined based on the original transition function of $\mathcal{A}$. When the final state in $\mathcal{B}$ is a strongly connected component C of $G'_{\mathcal{A}}$, $\mathcal{B}$ chooses a vertex v from C according to the stationary distribution π_C of a random walk in C, and then outputs what $\mathcal{A}$ outputs when its state is v. We note that when a stream σ is executed by $\mathcal{B}$, what essentially happens is that some zero-frequency streams (corresponding to moving along arcs in $G'_{\mathcal{A}}$) are inserted into σ to form a new stream σ' which will be executed by $\mathcal{A}$. We have that $\sigma \in \Lambda_m$ implies $\sigma' \in \Lambda_m$; but when $\sigma \in \Gamma_m$, the frequency of some prefix of σ' could be very large. Thus $\mathcal{A}$ is required to solve the problem on Λ_m to make the reduction work.

Define L to be the maximum length of the shortest zero-frequency path connecting a pair of vertices. Here the maximum is taken over all pairs of vertices that are connected by at least one zero-frequency path. Note that the zero-frequency streams inserted into σ correspond to taking a walk in the zero-frequency graph $G'_{\mathcal{A}}$. Thus we can assume that the lengths of inserted zero-frequency streams are at most L: this can be achieved by always choosing the shortest zero-frequency paths between pairs of vertices. Then it is easy to see that $\sigma \in \Gamma_{m/2}$ implies $\sigma' \in \Gamma_{m/2+L/2}$. Hence, if $L \leq m$, we will be able to perform the reduction from an initial algorithm for input streams in Γ_m to a linear sketch for input streams in $\Gamma_{m/2}$.

Note that the reduction we use here is the same as step (i) in the LNW reduction. We obtain a path-independent automaton *regardless* of what input streams we are considering. What changes is that we have argued, when $L \leq m$ is satisfied, the path-independent automaton we obtain will be correct on $\Gamma_{m/2}$ if the original automaton is correct on Γ_m. Now it remains to find a sufficient condition for $L \leq m$.

We will prove an upper bound on L in terms of n and the number of vertices $s = |V|$ in order to obtain a condition for removing the box constraint. The idea to upper bound L is to build linear equations based on the graph $G_{\mathcal{A}}$ such that the equations have a positive integer solution if and only if there exists a zero-frequency path from a given state to another. Then, Lemma 3.1 below enables us to find a positive integer solution of small magnitude if there exists one. Finally, we convert the bound on the magnitude of the solutions to the bound on the length of the path. In the LNW reduction, one way to obtain a finite bound on L as mentioned in that work is to build a system of linear equations in terms of simple paths and simple cycles, though an exact bound is not given in [9]. We note that an upper bound $s^{O(s+n)}$ on L can be obtained via this approach. (See Appendix B.) Unfortunately the bound $s^{O(s+n)}$ is not strong enough for our applications. Here in Lemma 3.3 we propose a better way to build the linear equations, which gives us a tighter bound of $\mathrm{poly}(sn) \cdot (\frac{s}{n}+1)^n$. Instead of writing the linear equations in terms of simple cycles, we write them in terms of arcs.

▶ **Lemma 3.1.** *Let A be an $m \times n$ integer matrix and $b \in \mathbb{Z}^m$. Suppose that M_1 is an upper bound on the absolute value of any sub-determinant of the matrix $\begin{pmatrix} A & b \end{pmatrix}$. If $Ax = b$ has a positive integer solution, then it has one whose all coordinates are at most $(n+1)^2 M_1$.*

Proof. We make use of a result in [12]. Let C be a $p \times n$ integer matrix and $d \in \mathbb{Z}^p$. Let r be the rank of A. Suppose that M is an upper bound on the absolute value of any sub-determinant of the matrix $\begin{pmatrix} A & b \\ C & d \end{pmatrix}$, which contains at least r rows from $\begin{pmatrix} A & b \end{pmatrix}$. The

[5] A strongly connected component is said to be terminal if there is no arc coming from it to the rest of the graph.

following upper bound is shown on the magnitude of an integer solution to the linear system $\{Ax = b, Cx \geq d\}$:

▶ **Lemma 3.2** ([12]). *If $Ax = b$ and $Cx \geq d$ have a common integer solution, then they have one whose coordinates have absolute values at most $(n+1)M$.*

Now we let $p = n$, $C = I_n$, and $d = \vec{1} = (1, \ldots, 1)^\top$ and invoke Lemma 3.2. Then we know that $Ax = b$ has a positive integer solution whose coordinates are at most $(n+1)M$, where M is an upper bound on the absolute value of any sub-determinant of the matrix $\begin{pmatrix} A & b \\ I_n & \vec{1} \end{pmatrix}$, which contains at least r rows from $\begin{pmatrix} A & b \end{pmatrix}$.

Then it suffices to prove $M \leq (n+1)M_1$. Consider an arbitrary submatrix T of $\begin{pmatrix} A & b \\ I_n & \vec{1} \end{pmatrix}$, which contains at least r rows from $\begin{pmatrix} A & b \end{pmatrix}$. Note that all entries of $\begin{pmatrix} I_n & \vec{1} \end{pmatrix}$ are in $\{0, 1\}$ and that there are $n+1$ ways to choose one non-zero entry from each row of $\begin{pmatrix} I_n & \vec{1} \end{pmatrix}$ such that no two entries are in the same column. Thus, after expanding the determinant of T along its rows from $\begin{pmatrix} I_n & \vec{1} \end{pmatrix}$, $\det(T)$ can be written as the sum of no more than $n+1$ sub-determinants (multiplied by ± 1) of $\begin{pmatrix} A & b \end{pmatrix}$. Therefore $|\det(T)|$ is at most $n+1$ times the largest absolute value of any sub-determinant of $\begin{pmatrix} A & b \end{pmatrix}$, which implies $M \leq (n+1)M_1$. ◀

▶ **Lemma 3.3.** *Let $s = |V|$ be the number of vertices in the transition graph $G_{\mathcal{A}} = (V, E)$. Then $L \leq 2ns(2ns+1)^2 \cdot (\frac{s}{n} + 1)^n$, where L is the maximum length of the shortest zero-frequency path between any two vertices connected by at least one zero-frequency path.*

Proof. Consider any two vertices $o_1, o_2 \in V$ such that there exists a zero-frequency path from o_1 to o_2. We fix a subset of edges $E' = \{(u_1, v_1), (u_2, v_2), \ldots, (u_t, v_t)\}$ satisfying the following condition: it is possible to use and only use $(u_1, v_1), \ldots, (u_t, v_t)$ to reach o_2 from o_1. For every possible E' satisfying this condition, we build a linear system as follows.

Let $x \in \mathbb{Z}_+^t$ be the variable whose i-th coordinate x_i represents the number of times the arc (u_i, v_i) occurs in the path from o_1 to o_2. We will need two types of constraints to write the linear equations: (1) the frequency of the path is $\vec{0}$; (2) for each node v, the number of times we go out from v minus the number of times we go into v is 1 if $v = o_1$, is -1 if $v = o_2$, and is 0 otherwise.[6] Then each positive integer solution x to the above constraints corresponds to a zero-frequency path from o_1 to o_2 using the arcs in E'. It is easy to see that the above constraints can be written as linear equations $Ax = b$, where $A = \begin{pmatrix} f_{(u_1,v_1)} & f_{(u_2,v_2)} & \cdots & f_{(u_t,v_t)} \\ e_{u_1} - e_{v_1} & e_{u_2} - e_{v_2} & \cdots & e_{u_t} - e_{v_t} \end{pmatrix}$ is an $(n+s) \times t$ matrix, and $b = \begin{pmatrix} \vec{0} \\ e_{o_1} - e_{o_2} \end{pmatrix} \in \mathbb{R}^{n+s}$. Here e_v is the standard unit column vector in $\mathbb{R}^s$ with the non-zero coordinate corresponding to node v. (Note that there are s nodes in total so we can map every node to a coordinate.) The upper n rows guarantee the frequency of the path is 0. Here, recall that $f_{(u_i,v_i)} \in \mathbb{R}^n$ is the positive or negative standard unit column vector which is the update corresponding to the arc (u_i, v_i). The lower s rows are the network flow constraints. Note that all entries in $\begin{pmatrix} A & b \end{pmatrix}$ are in $\{-1, 0, 1\}$.

Since there exists a zero-frequency path from o_1 to o_2, at least one such system of the linear equations (i.e., for at least one fixing of E') has a positive integer solution. Next, we consider such a fixing of E' that leads to a positive integer solution to the corresponding linear

[6] These are called the network flow constraints.

system. By Lemma 3.1, if there exists a positive integer solution to $Ax = b$, then there exists such a solution x satisfying $\|x\|_\infty \le (t+1)^2 M_1$, where M_1 is the largest possible absolute value of any sub-determinant of $\begin{pmatrix} A & b \end{pmatrix}$. This solution x corresponds to a zero-frequency path of length $\|x\|_1 \le t\|x\|_\infty \le t(t+1)^2 M_1$.

Let S be an arbitrary square submatrix of $\begin{pmatrix} A & b \end{pmatrix}$. We write S as $S = \begin{pmatrix} X \\ Y \end{pmatrix}$, where X comes from the top n rows and Y comes from the bottom s rows of $\begin{pmatrix} A & b \end{pmatrix}$. Let the size of X be $h \times w$, and the size of Y be $(w-h) \times w$. (Clearly, we have $h \le n$ and $w \le s + h$.) We expand $\det(S)$ along its rows in X. Since each column in X has at most one non-zero entry (± 1), the rows of X have support on disjoint subsets of columns, and thus X has at most w non-zero entries in total. Then, by the AM-GM inequality, the number of ways to choose one non-zero entry from each row of X such that no two of them are in the same column is at most $\left(\frac{w}{h}\right)^h \le \left(\frac{s+h}{h}\right)^h = \left(1 + \frac{s}{h}\right)^h \le \left(1 + \frac{s}{n}\right)^n$. Then we have that $|\det(S)|$ is at most $\left(1 + \frac{s}{n}\right)^n$ times the maximum absolute value of any sub-determinant of Y. Let $C = (c_{ij})_{k \times k}$ be any submatrix of Y, which is also a submatrix of $\begin{pmatrix} e_{u_1} - e_{v_1} & e_{u_2} - e_{v_2} & \dots & e_{u_t} - e_{v_t} & e_{o_1} - e_{o_2} \end{pmatrix}$.

We show that $\det(C) \in \{0, 1, -1\}$. Note that C has at most two non-zero entries in each column, and if a column of C has two non-zero entries, they must be 1 and -1. If all columns in C have two non-zero entries, then $(1, 1, \dots, 1) \cdot C = (0, 0, \dots, 0)$, which implies $\det(C) = 0$. If there exists a column in C without non-zero entries, then we also have $\det(C) = 0$. Otherwise we can find a column with exactly one non-zero entry c_{ij}, and then we have $\det(C) = (-1)^{i+j} c_{ij} \det(D_{ij})$, where D_{ij} is formed by deleting row i and column j from C. By induction, we have $\det(C) \in \{-1, 0, 1\}$. Therefore, we know that $|\det(S)| \le \left(1 + \frac{s}{n}\right)^n \cdot 1 = \left(1 + \frac{s}{n}\right)^n$.

Since S is arbitrary, we have $M_1 \le \left(1 + \frac{s}{n}\right)^n$. Note that there are at most $2ns$ arcs in the graph, so we have $t \le 2ns$. Then we can bound the length of a zero-frequency path by $t(t+1)^2 M_1 \le 2ns(2ns+1)^2 \left(1 + \frac{s}{n}\right)^n$. ◀

Let $r = |C(\mathcal{A}, m)|$. Note that we only care about the correctness of $\mathcal{A}$ on input streams from Γ_m. Thus we can, without loss of generality, combine all states not in $C(\mathcal{A}, m)$ (if there are any) into an "irreversible crash" state: the automaton will stay in this state after leaving $C(\mathcal{A}, m)$. This modification will not affect the correctness of $\mathcal{A}$ on Γ_m. Therefore, we can assume $s \le r + 1$. Then Lemma 3.3 implies $L \le 2n(r+1)(2n(r+1)+1)^2 \cdot \left(\frac{r+1}{n} + 1\right)^n$.

Assume $r > 1$. Then we have $L \le (4nr)^3 \cdot (r+2)^n \le r^{c_1 n}$ for a sufficiently large constant $c_1 > 0$. Recall that we want $L \le m$. Taking logarithms on both sides of the desired inequality $r^{c_1 n} \le m$, we equivalently want $c_1 n \log r \le \log m$, i.e., $\log r \le \frac{\log m}{c_1 n}$. Therefore, when we have $S(\mathcal{A}, m) = \log r \le \frac{\log m}{c_1 n}$ for some fixed constant $c_1 > 0$, the condition $L \le m$ will be satisfied. In this case, according to our analysis, the box constraint can be removed. Combining this condition and [9, Theorem 10] (with minor correction), we summarize our results on removing the box constraint as the following theorem.

▶ **Theorem 3.4** (Removing the box constraint). *Suppose that a randomized 1-pass streaming algorithm $\mathcal{A}$ solves a problem P on any stream in Γ_m with probability at least $1 - \delta$, and that the space used by any deterministic instance of $\mathcal{A}$ is no more than $c \cdot \frac{\log m}{n}$, where $c > 0$ is a universal constant. Then there exists an algorithm $\mathcal{B}$ implemented by maintaining a linear sketch $A \cdot x \bmod q$ in the stream, where A is a random $r \times n$ integer matrix and q is a random positive integer vector of length r, such that $\mathcal{B}$ solves P on any stream in $\Gamma_{m/2}$ with probability $1 - 6\delta$ and that $S(\mathcal{B}, m/2) \le S(\mathcal{A}, m) + O(\log n + \log\log m + \log\frac{1}{\delta})$.*[7]

[7] The extra $O(\log n + \log\log m + \log\frac{1}{\delta})$ bits are used only for the randomness for sampling A and q.

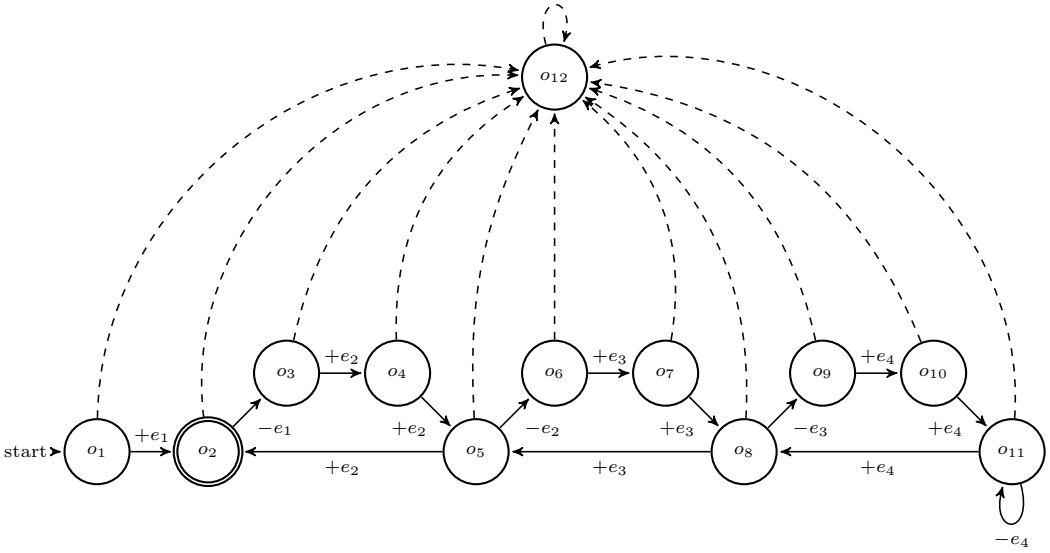


Figure 1 An illustrative example.

3.2 Tightness of Our Condition

In Lemma 3.3, we show that an upper bound of L is $\text{poly}(ns) \cdot (1 + \frac{s}{n})^{O(n)}$. Now we give an example of an automaton to show this upper bound is essentially tight (assuming s is much larger than n). In our example, there exist two vertices in the transition graph of the automaton so that the length of the shortest zero-frequency path between them is at least $(\frac{s}{n})^{\Omega(n)}$. The matching upper bound and lower bound on L eliminate the possibility of getting any further improvement in the condition for removing the box constraint, if the same reduction in [9] is used.

An illustrative example of our construction with $n = 4$ and $s = 12$ is given in Figure 1. Here each dashed arc represents all remaining outgoing arcs from a vertex. Consider the shortest zero-frequency path from o_1 to o_2. (Note that there exists one such path.) After going from o_1 to o_2 by the arc with $+e_1$ update, the path needs to go through the cycle $o_2 \to o_3 \to o_4 \to o_5 \to o_2$ one time in order to make the first coordinate of the frequency vector x equal to 0. However, that causes the second coordinate to be 3 and the path then needs to go through the cycle $o_5 \to o_6 \to o_7 \to o_8 \to o_5$ three times for compensation. That further causes the third coordinate to be 3^2 and the path needs to go through $o_8 \to o_9 \to o_{10} \to o_{11} \to o_8$ a total of 3^2 times. Finally, the path needs to go through the self-loop at o_{11} with $-e_4$ update a total of 3^3 times. Note that the number of times the shortest zero-frequency path from o_1 to o_2 passes through each cycle goes up exponentially.

▶ **Lemma 3.5.** *There exists an automaton $\mathcal{A}$ such that the transition graph $G_{\mathcal{A}} = (V, E)$ satisfies $L = (\frac{s}{n})^{\Omega(n)}$, where $s = |V|$ is the number of vertices, and L is the maximum length of the shortest zero-frequency path between any two vertices connected by at least one zero-frequency path.*

Proof. We assume $n > 3$, $s - 3 > 3(n-1)$ and $(n-1) \mid (s-3)$. For $(n-1) \nmid (s-3)$, we can decrease s until $(n-1) \mid (s-3)$ is satisfied. We construct a transition graph $G_{\mathcal{A}} = (V, E)$ whose structure is similar to the example in Figure 1. Let $V = \{o_1, o_2, \ldots, o_s\}$. There are $2n$ outgoing arcs from each of the vertices in V. We write $o_i \oplus \pm e_k = o_j$ to stand for an arc

(o_i, o_j) in E where $f_{(o_i,o_j)} = \pm e_k$. Let $len = (s-3)/(n-1)+1$. The arcs in E are defined as follows:

- $o_1 \oplus +e_1 = o_2$.
- $o_{s-1} \oplus -e_n = o_{s-1}$.
- For $i \equiv 2 \bmod (len-1)$ and $i \neq s-1$, $o_i \oplus -e_{\lceil i/(len-1)\rceil} = o_{i+1}$.
- For $i \equiv 2 \bmod (len-1)$ and $i \neq 2$, $o_i \oplus +e_{\lceil i/(len-1)\rceil} = o_{i-(len-1)}$.
- For $i \not\equiv 2 \bmod (len-1)$, $i \neq 1$ and $i \neq s$, $o_i \oplus +e_{\lfloor i/(len-1)\rfloor+1} = o_{i+1}$.
- For a vertex o_i and $e \in \{+e_1, \ldots, +e_n, -e_1, \ldots, -e_n\}$ where $o_i \oplus e$ is undefined from above, $o_i \oplus e = o_s$.

There are $n-1$ cycles each of which has length len. The i-th cycle consists of nodes $o_{(i-1)(len-1)+2}, o_{(i-1)(len-1)+3}, \ldots, o_{i(len-1)+2}$. Among the arcs in the i-th cycle, there is one arc with $-e_i$ update, and all other $len-1$ arcs are with $+e_{i+1}$ update. In this case, any zero-frequency path from o_1 to o_2 passes through the i-th cycle at least $(len-1)^{i-1}$ times, and thus its length is at least $\sum_{i=1}^{n-1} len \cdot (len-1)^{i-1} = len \cdot (len-1) \cdot \frac{(len-1)^{n-1}-1}{len-2} = (\frac{s}{n})^{\Omega(n)}$. ◀

In fact, for our example we can have a stronger statement that along any zero-frequency path from o_1 to o_2, some coordinate of the underlying vector achieves at least $(\frac{s}{n})^{\Omega(n)}$. This is the reason why the underlying vector could escape from $\{-m, -(m-1), \ldots, m\}^n$ at the middle of the stream after inserting zero-frequency streams in the reduction. In order to remove the box constraint, there must be some constants C_1, C_2 such that $(\frac{s}{n})^{C_1 n} \leq C_2 m$, i.e., $\log s \leq \frac{\log m + \log C_2}{C_1 n} + \log n$. When m is sufficiently large and n is fixed, this implies $\log s \leq \frac{C \log m}{n}$ for some constant C. Hence our condition for removing the box constraint is tight.

3.3 Space Lower Bounds for Additive Error Norm Approximation

We consider the problem of estimating the ℓ_p-norm $\|x\|_p (p > 0)$ in the turnstile streaming model, where the underlying vector x is promised to be in $\{-m, -(m-1), \ldots, m\}^n$ at all points in the stream (i.e., without the box constraint). We prove an $\Omega(\log m)$ bit space lower bound for approximating $\|x\|_p$ up to an additive $\epsilon n^{1/p} m$ error, where $\epsilon \in (0, \frac{1}{2})$ is a constant. Our proof makes use of the LNW reduction with the box constraint removed.

A Norm Decision Problem. First we consider the following promise problem: we are given the promise that the input $x \in \{-m, -(m-1), \ldots, m\}^n$ satisfies either $\|x\|_p \leq \alpha n^{1/p} m$ or $\|x\|_p \geq \beta n^{1/p} m$, where $0 < \alpha < \beta < 1$ are constants, and need to decide whether $\|x\|_p \leq \alpha n^{1/p} m$ or $\|x\|_p \geq \beta n^{1/p} m$. We first prove that this problem has an $\Omega(\log m)$ space lower bound.

▶ **Theorem 3.6** (Norm decision problem). *For any constants $p > 0$, $0 < \alpha < \beta < 1$ and $0 \leq \delta < \frac{\min\{\alpha, 1-\beta\}}{6(\alpha+1-\beta)}$, any 1-pass streaming algorithm which, for any input $x \in \{-m, -(m-1), \ldots, m\}^n$, decides whether $\|x\|_p \leq \alpha n^{1/p} m$ or $\|x\|_p \geq \beta n^{1/p} m$ (provided that x satisfies one of them) with probability at least $1-\delta$ in the turnstile model uses $\Omega(\log m)$ bits of space.*

Proof. We assume without loss of generality that $n = 1$, since one can always use an algorithm for larger n to solve the problem for $n = 1$ by assigning all n coordinates the same value. Suppose that the theorem does not hold. Then for *any* sufficiently small constant $\epsilon > 0$, there exists m and an algorithm $\mathcal{A}$ such that $\mathcal{A}$ uses less than $\epsilon \log m$ bits of space and solves the given problem (with parameters m and $n = 1$) with probability $1-\delta$. Since the space used by $\mathcal{A}$ is less than $\frac{\epsilon \log m}{n}$ bits (using $n = 1$), from Theorem 3.4 we know that there

is an algorithm $\mathcal{B}$ that maintains a linear sketch $A \cdot x \bmod q$ and solves the same problem[8] with probability $1 - 6\delta$. Furthermore, the space used by any deterministic instance of $\mathcal{B}$ is also less than $\epsilon \log m$ bits.

Let $U_1 = \{0, 1, \ldots, \lfloor \alpha m \rfloor\}$ and $U_2 = \{\lceil \beta m \rceil, \lceil \beta m \rceil + 1, \ldots, m\}$. Let Π be the uniform distribution on $U = U_1 \cup U_2$. By Yao's minimax principle, there exists a fixing of A and q that solves the problem for x drawn from Π (i.e., decides if x is in U_1 or U_2) with probability at least $1 - 6\delta$. Since $n = 1$, we can write $Ax \bmod q = (a_1 x \bmod q_1, a_2 x \bmod q_2, \ldots, a_r x \bmod q_r)$, where $a_1, \ldots, a_r \in \mathbb{Z}$ and $q_1, \ldots, q_r \in \mathbb{Z}_+$. Without loss of generality, we assume $\gcd(a_i, q_i) = 1$ for $i = 1, \ldots, r$. (If $\gcd(a_i, q_i) = d > 1$, we can let $a_i' = a_i/d, q_i' = q_i/d$ and then there is a one-to-one correspondence between $a_i x \bmod q_i$ and $a_i' x \bmod q_i'$. So a_i and q_i can be replaced by a_i' and q_i'.) Let $l = \operatorname{lcm}(q_1, \ldots, q_r)$.

We now prove $l = \Omega(m)$. Suppose that $l < \min\{\alpha, 1 - \beta\} \cdot m$ (otherwise we already have $l = \Omega(m)$). The input space U can be partitioned into l groups $G_i = \{j \in U \big| (i - j) \bmod l = 0\} (i = 0, 1, \ldots, l - 1)$. Note that the algorithm outputs the same answer for inputs from the same group. Within group G_i, the algorithm outputs the correct answer for at most a $\frac{\max\{|G_i \cap U_1|, |G_i \cap U_2|\}}{|G_i|}$ fraction of inputs. For $i \in \{0, 1, \ldots, l - 1\}$ we have $|G_i \cap U_1| = \lfloor \frac{\alpha m - i}{l} \rfloor + 1 \in (\frac{\alpha m}{l} - 1, \frac{\alpha m}{l} + 1]$ and $|G_i \cap U_2| = \lfloor \frac{m - i}{l} \rfloor - \lceil \frac{\beta m - i}{l} \rceil + 1 \in (\frac{(1-\beta)m}{l} - 1, \frac{(1-\beta)m}{l} + 1]$. Thus

$$\frac{\max\{|G_i \cap U_1|, |G_i \cap U_2|\}}{|G_i|} \leq \frac{\max\{\frac{\alpha m}{l} + 1, \frac{(1-\beta)m}{l} + 1\}}{(\frac{\alpha m}{l} - 1) + (\frac{(1-\beta)m}{l} - 1)} = \frac{\max\{\alpha, 1 - \beta\} \frac{m}{l} + 1}{(\alpha + 1 - \beta)\frac{m}{l} - 2}.$$

The above is an upper bound of the success probability on Π, so we must have $1 - 6\delta \leq \frac{\max\{\alpha, 1-\beta\} \frac{m}{l} + 1}{(\alpha+1-\beta)\frac{m}{l} - 2}$, which means $l \geq \frac{(1-6\delta)(\alpha+1-\beta) - \max\{\alpha, 1-\beta\}}{3 - 12\delta} m$. Since $\delta < \frac{\min\{\alpha, 1-\beta\}}{6(\alpha+1-\beta)}$, we have $(1 - 6\delta)(\alpha + 1 - \beta) - \max\{\alpha, 1 - \beta\} > 0$. Therefore $l = \Omega(m)$.

Next we show that as x varies in $\{1, 2, \ldots, l\}$, $A \cdot x \bmod q$ takes l distinct values. Suppose that there are $x, y \in \{1, 2, \ldots, l\} (x \neq y)$ such that $A \cdot x \bmod q = A \cdot y \bmod q$. Then for all $i \in \{1, \ldots, r\}$ we have $a_i(x - y) \bmod q_i = 0$, which means $(x - y) \bmod q_i = 0$ since $\gcd(a_i, q_i) = 1$. Therefore $(x - y) \bmod (\operatorname{lcm}(q_1, \ldots, q_r)) = 0$, i.e., $(x - y) \bmod l = 0$, a contradiction. So $A \cdot x \bmod q$ takes l distinct values as x varies in $\{1, 2, \ldots, l\}$. This means that as x varies in $\{1, 2, \ldots, m\}$, $A \cdot x \bmod q$ takes $\min\{m, l\}$ distinct values, so the space complexity of maintaining $A \cdot x \bmod q$ is at least $\Omega(\log(\min\{m, l\})) = \Omega(\log m)$, which is a contradiction. ◀

The following $\Omega(\log m)$ lower bounds are corollaries of Theorem 3.6.

▶ **Theorem 3.7** (Additive error norm approximation). *For any constants $p > 0$ and $0 \leq \epsilon < \frac{1}{2}$, any 1-pass streaming algorithm which, for any input $x \in \{-m, -(m-1), \ldots, m\}^n$, outputs an approximation of $\|x\|_p$ in the interval $\left[\|x\|_p - \epsilon n^{1/p} m, \|x\|_p + \epsilon n^{1/p} m\right]$ with probability greater than $\frac{11}{12}$ in the turnstile model uses $\Omega(\log m)$ bits of space.*

Proof. Suppose that the theorem does not hold. Then for *any* sufficiently small constant $\eta > 0$, there exists m and an algorithm $\mathcal{A}$ such that $\mathcal{A}$ uses less than $\eta \log m$ bits of space and estimate $\|x\|_p$ (for any $x \in \{-m, -(m-1), \ldots, m\}^n$) up to additive $\epsilon n^{1/p} m$ error with probability $1 - \delta$, where $0 \leq \delta < \frac{1}{12}$. Below, we make use of $\mathcal{A}$ to solve the norm decision problem in $\eta \log m$ bits of space and thus reach a contradiction to Theorem 3.6.

[8] According to Theorem 3.4, $\mathcal{B}$ can only solve the problem with parameter $m/2$ instead of m. Since we are proving an $\Omega(\log m)$ lower bound, we can replace $m/2$ by m for simplicity.

We invoke $\mathcal{A}$ to solve the norm decision problem with parameters $\alpha < \frac{1}{2} - \epsilon$, $\beta > \frac{1}{2} + \epsilon$, and δ. We can choose α and β such that $\alpha = 1 - \beta$, then we have $\delta < \frac{1}{12} = \frac{\min\{\alpha, 1-\beta\}}{6(\alpha+1-\beta)}$ as required in Theorem 3.6. When $\|x\|_p \leq \alpha n^{1/p} m$, a successful estimate for $\|x\|_p$ given by $\mathcal{A}$ will be at most $(\alpha + \epsilon)n^{1/p}m$; when $\|x\|_p \geq \beta n^{1/p} m$, a successful estimate for $\|x\|_p$ given by $\mathcal{A}$ will be at least $(\beta - \epsilon)n^{1/p}m$. Since $\alpha + \epsilon < \frac{1}{2} < \beta - \epsilon$, by looking at the most significant $O(1)$ bits of the output of $\mathcal{A}$, we are able to tell whether the output is at most $(\alpha + \epsilon)n^{1/p}m$ or at least $(\beta - \epsilon)n^{1/p}m$, and thus to decide whether $\|x\|_p \leq \alpha n^{1/p} m$ or $\|x\|_p \geq \beta n^{1/p} m$ (with probability at least $1 - \delta$). This solves the norm decision problem using $\eta \log m$ bits of space, contradicting Theorem 3.6. ◀

▶ **Theorem 3.8** (Approximating a counter up to additive error). *For any constant $0 \leq \epsilon < \frac{1}{2}$, any 1-pass algorithm which, for any input $x \in \{-m, -(m-1), \ldots, m\}$, outputs $|x|$ up to additive ϵm error with probability larger than $\frac{11}{12}$ in the turnstile model uses $\Omega(\log m)$ bits of space.*

Proof. This is a special case of Theorem 3.7 with $n = 1$. ◀

▶ **Theorem 3.9** (Maintaining the most significant bit of a counter). *Any 1-pass algorithm which, for any input $x \in \{-m, -(m-1), \ldots, m\}$, outputs the most significant bit of $|x|$ with probability larger than $\frac{11}{12}$ in the turnstile model uses $\Omega(\log m)$ bits of space.*

Proof. Without loss of generality, we assume $m = 2^k - 1 (k \in \mathbb{Z}_+)$. In this case $|x|$ has k bits. If an algorithm can output the most significant bit of $|x|$, it must be able to distinguish whether $|x| \leq \frac{1}{4}m$ or $|x| \geq \frac{3}{4}m$: the most significant bit of $|x|$ is 0 in the former case, and is 1 in the latter case. Then the $\Omega(\log m)$ lower bound follows from Theorem 3.6. ◀

4 Reduction to Linear Sketches in the Strict Turnstile Model

In this section, we show that in the strict turnstile model, there is also an equivalence between general algorithms and linear sketches, similar to the LNW reduction for the turnstile model. We will consider algorithms that allow input streams from Λ_m^*, which is the set of streams such that the underlying vector never has a negative entry and is in $\{0, 1, \ldots, m\}^n$ at the end of the stream. We further assume that the algorithms have space complexity depending only on the dimension n, which is suitable when we want to prove lower bounds as functions of n, such as in graph problems.

The following theorem is an adaptation of [9, Theorem 10] for the strict turnstile model. It implies that to obtain lower bounds depending only on dimension in the strict turnstile model, it suffices to consider linear sketches, or the simultaneous communication model.

▶ **Theorem 4.1** (Reduction in the strict turnstile model). *Suppose that a randomized algorithm $\mathcal{A}$ solves a problem P on any stream in Λ_m^* with probability at least $1 - \delta$, and that the space complexity of $\mathcal{A}$ depends only on n. Then there exists an algorithm $\mathcal{B}$ implemented by maintaining a linear sketch $A \cdot x \bmod q$ in the stream, where A is a random $r \times n$ integer matrix and q is a positive integer vector of length r, such that $\mathcal{B}$ solves P on any stream in Λ_m^* with probability at least $1 - 6\delta$ and that the space used by any deterministic instance of $\mathcal{B}$ is no more than the space used by $\mathcal{A}$.*

Proof. We modify the reduction from general automaton to path-independent automaton (which was only claimed to be path-reversible in [9], as mentioned in the beginning of this section).

We view $\mathcal{A}$ as an automaton. Since the space used by $\mathcal{A}$ depends only on n, there is a function g such that the number of states of every deterministic instance of $\mathcal{A}$ is no more than $g(n)$. As in Section 3.1, let L be the maximum length of the shortest zero-frequency path between any two states in the transition graph of any deterministic instance of $\mathcal{A}$. From Lemma 3.3 we know that $L \leq h(n)$ for some function h.

Let γ be a fixed stream with frequency $(h(n), \ldots, h(n))$, which consists of only positive updates (i.e., $+e_i$'s). We construct another automaton $\mathcal{A}'$ as follows: for any randomness, (1) $\mathcal{A}'$ has the same transition graph as $\mathcal{A}$; (2) the starting state of $\mathcal{A}'$ is $o \oplus \gamma$, where o is the starting state of $\mathcal{A}$; (3) for any state u, the output of $\mathcal{A}'$ on u is the output of $\mathcal{A}$ on the state $u \oplus \gamma^{-1}$.

It is easy to see that executing a stream σ on $\mathcal{A}'$ is equivalent to running the stream $\gamma \circ \sigma \circ \gamma^{-1}$ on $\mathcal{A}$. Since $\mathcal{A}$ succeeds in solving P on any stream in Λ_m^* with probability at least $1 - \delta$, we know that $\mathcal{A}'$ solves P on a stream σ with probability $1 - \delta$ as long as $\gamma \circ \sigma \circ \gamma^{-1} \in \Lambda_m^*$, i.e., $\mathcal{A}'$ solves P on any stream in the set $\Phi = \{\|\text{freq } \sigma\|_\infty \leq m, \text{freq } \sigma \geq \vec{0},$ and the frequency of any prefix of σ has all its coordinates at least $-h(n)\}$ with probability $1 - \delta$.

Now we invoke the LNW reduction from $\mathcal{A}'$ to a path-reversible automaton $\mathcal{C}$. According to Appendix A, $\mathcal{C}$ is as well a path-independent automaton. Recall that when a stream $\sigma \in \Lambda_m^*$ is executed by $\mathcal{C}$, equivalently another stream σ' is executed by $\mathcal{A}'$, where σ' is obtained by inserting zero-frequency streams into σ. Note that we can assume that all the inserted zero-frequency streams have length at most $L \leq h(n)$, and then $\sigma \in \Lambda_m^*$ implies $\sigma' \in \Phi$. Therefore we have a path-independent automaton $\mathcal{C}$ solving P on any stream in Λ_m^* (with high probability). ◀

Maximum Matching. In a recent work [2], tight upper and lower bounds are shown for turnstile algorithms that approximate maximum matching in dynamic graph streams, but the lower bound is only proved in the simultaneous communication model. Using Theorem 4.1, we are able to conclude that their results hold for any 1-pass algorithm and thus to resolve the 1-pass space complexity of this problem. Namely, to compute an n^ϵ-approximate maximum matching, $\Theta(n^{2-3\epsilon})$ bits of space is both sufficient and necessary (up to polylogarithmic factors), where n is the number of vertices.

5 Reduction for Multi-Pass Automata

Our main result in this section is that the LNW reduction can be extended to multi-pass automata, i.e., that a randomized p-pass automaton can be reduced to a path-independent one without blowing up the space complexity. Throughout this section p is a constant.

The main difficulty of this reduction is that when we consider an automaton in the i-th pass, for $i > 1$, we have to restrict to a subset of input streams that lead to the same state in the automaton processing the previous pass. Fortunately there is still sufficient randomness remaining even with this restriction so that the padding argument with zero-frequency streams in the LNW reduction still works.

▶ **Theorem 5.1** (Reduction of p-pass automata). *Let $\mathcal{A}$ be a p-pass randomized automaton that solves P with probability $\geq 1 - \delta$. Let $\epsilon > 0$. For any distribution Π over streams, there exists a p-pass path-independent deterministic automaton $\mathcal{B}$ that solves P over input drawn from Π with probability $\geq 1 - \delta - \epsilon$. Furthermore, $S(\mathcal{B}, m) \leq S(\mathcal{A}, m)$.*

Proof for $p = 2$. We first give a detailed proof for the case $p = 2$. The same method can be easily generalized to larger p.

Let S_0 be a set of zero-frequency streams such that whenever o_1 and o_2 are two states of $\mathcal{A}^1$ or of any automaton $\mathcal{A}^2_s$, there exists $\sigma \in S_0$ such that $o_1 \oplus \sigma = o_2$, where $\oplus$ is the transition function of the corresponding automaton.

Define a distribution Π' as follows. For a stream $\sigma \sim \Pi$ and $\sigma_0 = (\sigma_1, \ldots, \sigma_{2W}) \sim \mathrm{Unif}(S_0^{2W})$, we include $\sigma \otimes \sigma_0 := \sigma_1 \circ \ldots \circ \sigma_W \circ \sigma \circ \sigma_{W+1} \ldots \circ \sigma_{2W}$ in Π'. Here for any set S, $\mathrm{Unif}(S)$ is defined to be the uniform distribution over S. We shall choose W to be sufficiently large so that certain conditions are satisfied. The conditions will be described explicitly later in the proof.

By Yao's minimax principle, we can pick a deterministic instance of $\mathcal{A}$, also denoted by $\mathcal{A}$, which is correct on input distribution Π' with probability $\geq 1 - \delta$. Henceforth in the proof $\mathcal{A}$ refers to this deterministic instance. Everything is similar to [9] so far.

For a state s of $\mathcal{A}^1$, denote by $\Pi'(s)$ the marginal distribution of Π' on the event that reading the input stream in $\mathcal{A}^1$ ends at state s. By the correctness assumption of $\mathcal{A}$, it holds that

$$\mathop{\mathbb{E}}_{\sigma' \in \Pi'} \mathbf{1}\left\{\phi_{\mathcal{A}^2_{\sigma'}}(\sigma') \text{ is acceptable for } \sigma'\right\} \geq 1 - \delta,$$

or, equivalently,

$$\mathop{\mathbb{E}}_{s \sim \mu} \mathop{\mathbb{E}}_{\sigma' \sim \Pi'(s)} \mathbf{1}\left\{\phi_{\mathcal{A}^2_{\sigma'}}(\sigma') \text{ is acceptable for } \sigma'\right\} \geq 1 - \delta,$$

where μ the distribution over the states of $\mathcal{A}^1$ induced by Π'.

For each second-pass automaton $\mathcal{A}^2_s$ (s is a state in $\mathcal{A}^1$), we reduce it to a path-independent automaton $\mathcal{B}^2_s$ with the same transition functions as in the LNW reduction. Since $\mathcal{A}^2_s$ will only be run on the input streams in $\mathrm{supp}(\Pi'(s))$, we may assume that the states of $\mathcal{B}^2_s$ are all the terminal equivalence classes $\langle \tau' \rangle$ of $\mathcal{A}^2_s$, where $\tau' \in \mathrm{supp}(\Pi'(s))$.

To specify the output on the terminal equivalence class, we need the following proposition, whose proof is postponed to the end of this section.

▶ **Proposition 5.2.** *Let $\epsilon > 0$ and W be a sufficiently large integer. Suppose that $\sigma = \sigma_1 \circ \ldots \circ \sigma_W \sim \mathrm{Unif}(S_0^W)$ and C is a terminal equivalence class of some automaton. Let s_0 and s be arbitrary states in C and let event $\mathcal{E} = \{s_0 \oplus \sigma = s\}$. There exist a positive integer $L \leq W$ and a distribution $\mathcal{D}$ over S_0^{W-L} (both L and $\mathcal{D}$ are independent of s_0 and s) such that*

$$d_{TV}(\mathcal{L}(\sigma_{L+1} \circ \ldots \circ \sigma_W | \mathcal{E}), \mathcal{D}) \leq \epsilon,$$

where $\mathcal{L}(\sigma_{L+1} \circ \ldots \circ \sigma_W | \mathcal{E})$ is the conditional distribution of $\sigma_{L+1} \circ \ldots \circ \sigma_W$ on the event $\mathcal{E}$. Furthermore, there exists an integer $R \in [L, W]$ independent of s_0 and s such that $d_{TV}(\mathcal{L}(\sigma_{L+1} \circ \ldots \circ \sigma_R | \mathcal{E}), \mathrm{Unif}(S_0^{R-L})) \leq \epsilon$, and $R - L$ can be made arbitrarily large.

Now we specify the output of $\mathcal{B}^2_s$. Let $\langle t \rangle$ be a terminal class of $\mathcal{A}^2_s$ and $\Pi'(s, \langle t \rangle)$ be the marginal distribution of $\Pi'(s)$ on the streams terminating in $\langle t \rangle$. The random output on $\langle t \rangle$ is defined as $\phi_{\mathcal{A}^2_s}(\tau')$ with $\tau' \sim \Pi'(s, \langle t \rangle)$.

For a stream prefix ρ, we denote by $\Pi'(s, \rho)$ the marginal distribution of $\Pi'(s)$ on the streams with prefix ρ.

We choose W *sufficiently large* such that the following two conditions hold.

(A) With probability $\geq 1 - \epsilon$ over $\sigma' \sim \Pi'$, the second-pass automaton $\mathcal{A}^2_{\sigma'}$ arrives at a state in a terminal equivalence class on input stream σ'.

(B) Conditioned on (A), for any second-pass automaton $\mathcal{A}_s^2$, and for any stream prefixes ρ_1 and ρ_2 of streams in $\mathrm{supp}(\Pi'(s))$ that satisfy (i) ρ_i has the form $\sigma_1 \circ \ldots \circ \sigma_W \circ \sigma \circ \sigma_{W+1} \circ \ldots \circ \sigma_{W+n_i}$ for some $0 \le n_i \le W$ $(i = 1, 2)$ and (ii) ρ_1 and ρ_2 arrive in the same equivalence class of $\mathcal{A}_s^2$, the induced distribution on the terminating states of streams in $\Pi'(s, \rho_1)$ and that on the terminating states of streams in $\Pi'(s, \rho_2)$ are ϵ-close in total variation distance.

Condition (A) is possible by Proposition 5.2, which indicates that there is a sufficiently long random walk in $\mathcal{A}_s^2$, so starting from a node outside any terminating equivalence class, it will arrive at a state in a terminal equivalence class with a high probability; then take a union bound. We shall be conditioned on (A). Condition (B) is possible again because of Proposition 5.2. The streams with prefix ρ_1 and the streams with prefix ρ_2 will be close to $\mathrm{Unif}(S_0^L)$ on a segment of length L (where L can be made arbitrarily large) so they will first mix in the terminal equivalence class and the streams have similar distribution afterwards. Therefore, the induced distribution on the terminating states of streams in $\Pi'(s, \rho)$ is close to that induced by $\Pi'(s, \langle\rho\rangle)$.

Furthermore, when W is large enough, the random zero-padding $\sigma_1 \circ \ldots \circ \sigma_W$ before $\sigma \sim \Pi$ always leads to a state in a (random) equivalence class in $\mathcal{A}_s^2$. Choose the initial state of $\mathcal{B}_s^2$ according to the induced distribution on the terminal equivalence classes by streams drawn from $\Pi'(s)$. It follows that on reading $\sigma' \sim \Pi'$, the distribution on the terminating equivalence classes in $\mathcal{A}_{\sigma'}^2$ is ϵ-close to the distribution on the corresponding states in $\mathcal{B}_{\sigma'}^2$. They are not necessarily the same distribution because we choose a random initial state in $\mathcal{B}_s^2$. This is called the terminal class property.

To show the correctness of $\mathcal{B}$, we need to show (we may rescale ϵ if necessary)

$$\mathop{\mathbb{E}}_{\sigma\sim\Pi} \mathop{\mathbb{E}}_{\text{randomness of } \mathcal{B}} \mathbf{1}_{\{\phi_{\mathcal{B}}(\sigma) \text{ is acceptable for } \sigma\}} \ge 1 - \delta - O(\epsilon). \tag{1}$$

We have

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\sigma\sim\Pi} \mathop{\mathbb{E}}_{\text{randomness of } \mathcal{B}} \mathbf{1}_{\{\phi_{\mathcal{B}}(\sigma) \text{ is acceptable for } \sigma\}} \\
=\ &\mathop{\mathbb{E}}_{\sigma\sim\Pi} \mathop{\mathbb{E}}_{s\sim\text{Stationary}(\langle\sigma\rangle)} \mathop{\mathbb{E}}_{\text{randomness of } \mathcal{B}_s^2} \mathbf{1}_{\{\phi_{\mathcal{B}_s^2}(\sigma) \text{ is acceptable for } \sigma\}} \\
\ge\ &\mathop{\mathbb{E}}_{\sigma\sim\Pi} \mathop{\mathbb{E}}_{\sigma_0\sim\mathrm{Unif}(S_0^{2W})} \mathop{\mathbb{E}}_{\text{randomness of } \mathcal{B}_{\sigma\otimes\sigma_0}^2} \mathbf{1}_{\left\{\phi_{\mathcal{B}_{\sigma\otimes\sigma_0}^2}(\sigma) \text{ is acceptable for } \sigma\right\}} - \epsilon \\
\ge\ &\mathop{\mathbb{E}}_{\sigma\sim\Pi} \mathop{\mathbb{E}}_{\sigma_0\sim\mathrm{Unif}(S_0^{2W})} \mathop{\mathbb{E}}_{\tau'\sim\Pi'(\sigma\otimes\sigma_0,\langle\sigma\otimes\sigma_0\rangle)} \mathbf{1}_{\left\{\phi_{\mathcal{A}_{\sigma\otimes\sigma_0}^2}(\tau') \text{ is acceptable for } \sigma\right\}} - 2\epsilon.
\end{aligned} \tag{2}$$

In the above, line 2 follows from the random output of $\mathcal{B}^1$, line 3 from the fact that $\sigma \otimes \sigma_0$ with σ_0 is ϵ-close to the stationary distribution on $\langle\sigma\rangle$, line 4 from the definition of the random output of $\mathcal{B}^2$ and the terminal class property.

The event of the indicator function in (2) has a slight mismatch: the input stream is τ' while we only know the automaton's correctness on σ. To overcome this, we break up the streams in $\mathrm{supp}(\Pi'(\sigma \otimes \sigma_0, \langle\sigma \otimes \sigma_0\rangle))$ according to the frequency vectors v. We say $\mathrm{freq}(\tau')$ is admissible for $\tau' \in \mathrm{supp}(\Pi'(\sigma \otimes \sigma_0, \langle\sigma \otimes \sigma_0\rangle))$. Further conditioned on admissible v, the conditional distribution of $\Pi'(\sigma \otimes \sigma_0, \langle\sigma \otimes \sigma_0\rangle)$ is denoted by $\Pi'(\sigma \otimes \sigma_0, \langle\sigma \otimes \sigma_0\rangle, v)$. By condition (B), when W is sufficiently large, for any admissible v, the distribution of states in $\langle\sigma \otimes \sigma_0\rangle$ induced by $\Pi'(\sigma \otimes \sigma_0, \langle\sigma \otimes \sigma_0\rangle, v)$ is close to that induced by $\Pi'(\sigma \otimes \sigma_0, \langle\sigma \otimes \sigma_0\rangle)$. It therefore holds that

$$\left| \mathop{\mathbb{E}}_{\tau'\sim\Pi'(\sigma\otimes\sigma_0,\langle\sigma\otimes\sigma_0\rangle)} f(\tau') - \mathop{\mathbb{E}}_{\tau'\sim\Pi'(\sigma\otimes\sigma_0,\langle\sigma\otimes\sigma_0\rangle,\mathrm{freq}(\sigma))} f(\tau') \right| \le \epsilon, \tag{3}$$

for all (measurable) f with $\|f\|_\infty \le 1$. Now we can continue from (2):

$$
\begin{aligned}
&\mathop{\mathbb{E}}_{\sigma\sim\Pi}\ \mathop{\mathbb{E}}_{\text{randomness of }\mathcal{B}} \mathbf{1}_{\{\phi_{\mathcal{B}}(\sigma) \text{ is acceptable for } \sigma\}} \\
&\ge \mathop{\mathbb{E}}_{\sigma\sim\Pi}\ \mathop{\mathbb{E}}_{\sigma_0\sim\mathrm{Unif}(S_0^{2W})}\ \mathop{\mathbb{E}}_{\tau'\sim\Pi'(\sigma\otimes\sigma_0,\langle\sigma\otimes\sigma_0\rangle,\mathrm{freq}(\sigma))} \mathbf{1}_{\left\{\phi_{\mathcal{A}^2_{\sigma\otimes\sigma_0}}(\tau') \text{ is acceptable for } \sigma\right\}} - 3\epsilon \text{ (using (3))} \\
&= \mathop{\mathbb{E}}_{\sigma'\sim\Pi'}\ \mathop{\mathbb{E}}_{\tau'\sim\Pi'(\sigma',\langle\sigma'\rangle,\mathrm{freq}(\sigma'))} \mathbf{1}_{\left\{\phi_{\mathcal{A}^2_{\sigma'}}(\tau') \text{ is acceptable for } \tau'\right\}} - 3\epsilon \text{ (correctness depends only on freq. vec.)} \\
&= \mathop{\mathbb{E}}_{s\sim\mu}\ \mathop{\mathbb{E}}_{\sigma'\sim\Pi'(s)}\ \mathop{\mathbb{E}}_{\tau'\sim\Pi'(s,\langle\sigma'\rangle,\mathrm{freq}(\sigma'))} \mathbf{1}_{\left\{\phi_{\mathcal{A}^2_s}(\tau') \text{ is acceptable for } \tau'\right\}} - 3\epsilon \\
&\ge \mathop{\mathbb{E}}_{s\sim\mu}\ \mathop{\mathbb{E}}_{\sigma'\sim\Pi'(s)}\ \mathop{\mathbb{E}}_{\tau'\sim\Pi'(s,\langle\sigma'\rangle)} \mathbf{1}_{\left\{\phi_{\mathcal{A}^2_s}(\tau') \text{ is acceptable for } \tau'\right\}} - 4\epsilon \text{ (using (3))} \\
&= \mathop{\mathbb{E}}_{s\sim\mu}\ \mathop{\mathbb{E}}_{\tau'\sim\Pi'(s)} \mathbf{1}_{\left\{\phi_{\mathcal{A}^2_s}(\tau') \text{ is acceptable for } \tau'\right\}} - 4\epsilon \\
&\ge 1 - \delta - 4\epsilon.
\end{aligned}
$$

Removing the conditioning on (A) causes a further loss of ϵ in the success probability, that is,

$$\mathop{\mathbb{E}}_{\sigma\sim\Pi}\ \mathop{\mathbb{E}}_{\text{randomness of }\mathcal{B}} \mathbf{1}_{\{\phi_{\mathcal{B}}(\sigma) \text{ is acceptable for } \sigma\}} \ge 1 - \delta - 5\epsilon.$$

This completes the proof of (1).

Finally, by an averaging argument, there exists a deterministic automaton $\mathcal{B}$ achieving success probability at least as high as that of the randomized $\mathcal{B}$. The claim of space complexity follows from the same argument as in [9]. ◀

Proof of Theorem 5.1 for general p. We only describe the major changes on the proof for the special case $p = 2$. Here we choose W sufficiently large such that:

- **(A)** With probability $\ge 1 - \Theta(\epsilon)$ over $\sigma' \sim \Pi'$, the automaton $\mathcal{A}^q_{\sigma'}$ for all $2 \le q \le p$ arrives at a state in a terminal equivalence class on input stream σ'.
- **(B)** Over $\sigma' \sim \Pi'$, for any q-th pass automaton $\mathcal{A}^q_{\sigma'}$ $(2 \le q \le p)$, the induced distribution on terminating states of $\mathcal{A}^q_{\sigma'}$ is $\Theta(\epsilon)$-close to that on corresponding states of $\mathcal{B}^2_{\sigma'}$ on reading σ'. This is the terminal class condition.
- **(C)** (Conditioned on (A)) For any q-th pass automaton $\mathcal{A}^q_s$, and for any stream prefixes ρ_1 and ρ_2 of streams ending at the same state s of $\mathcal{A}^{q-1}$ such that both ρ_1 and ρ_2 arrive in the same equivalence class of $\mathcal{A}^q_s$, the induced distribution on the terminating states of streams with prefix ρ_1 and that on the terminating states of streams with prefix ρ_2 are $\Theta(\epsilon)$-close in total variation distance.

The random output is drawn from the stationary distribution on the associated equivalence class for the first-pass automaton, and is, for all subsequent passes $i \ge 2$, drawn from the induced distribution on the states of $\mathcal{A}^i_s$ by the conditional distribution of the streams terminating at s in the automaton of pass $i - 1$. A similar argument to that in the case of $p = 2$ gives the result. ◀

After we have Theorem 5.1, similar to [9], we can use Yao's minimax principle to conclude the existence of a randomized p-pass automaton that succeeds with probability $\ge 1 - \delta - \epsilon$ on any input, and can further use Newman's argument [11] to reduce the number of random bits to $O(\log n + \log\log m + \log\frac{1}{\epsilon})$.

Now we give the proof of Proposition 5.2.

Proof of Proposition 5.2. Let π be the stationary distribution on C under the transition probability induced by $\mathrm{Unif}(S_0)$. Note that

$$\begin{aligned}
&\Pr\{\sigma_{L+1}\circ\ldots\circ\sigma_W=\tau|s_0\oplus\sigma_1\circ\ldots\circ\sigma_W=s\}\\
=&\sum_{t\in C}\Pr\{\sigma_{L+1}\circ\ldots\circ\sigma_W=\tau|t\oplus\sigma_{L+1}\circ\ldots\circ\sigma_W=s,s_0\oplus\sigma_1\circ\ldots\circ\sigma_L=t\}\cdot\\
&\qquad\Pr\{s_0\oplus\sigma_1\circ\ldots\circ\sigma_L=t\}\\
=&\sum_{t\in C}\Pr\{\sigma_{L+1}\circ\ldots\circ\sigma_W=\tau|t\oplus\sigma_{L+1}\circ\ldots\circ\sigma_W=s\}\cdot\Pr\{s_0\oplus\sigma_1\circ\ldots\circ\sigma_L=t\}\\
=&\sum_{t\in C}\Pr\{\sigma_{L+1}\circ\ldots\circ\sigma_W=\tau|t\oplus\sigma_{L+1}\circ\ldots\circ\sigma_W=s\}\cdot\left(\pi(t)\pm\frac{\epsilon}{|C|}\right),
\end{aligned}$$

where line 3 follows from the Markov property of the process, line 4 follows from the fact that L can be chosen large enough so that $s_0\oplus\sigma_1\circ\ldots\circ\sigma_L$ mixes on C. Furthermore, L can be chosen independent of s_0 because there are only finitely many distinct s_0's. Define the probability distribution $\mathcal{D}$ as

$$\begin{aligned}
&\Pr_{\sigma_{L+1}\circ\ldots\circ\sigma_W\sim\mathcal{D}}\{\sigma_{L+1}\circ\ldots\circ\sigma_W=\tau\}\\
&\qquad=\mathop{\mathbb{E}}_{t\sim\pi}\Pr_{\sigma_{L+1}\circ\ldots\circ\sigma_W\sim\mathrm{Unif}(S_0^{W-L})}\{\sigma_{L+1}\circ\ldots\circ\sigma_W=\tau|t\oplus\sigma_{L+1}\circ\ldots\circ\sigma_W=s\}.
\end{aligned}$$

It is easy to verify that $\mathcal{D}$ is indeed a probability distribution. It follows that

$$\begin{aligned}
&d_{TV}(\mathcal{L}(\sigma_{L+1}\circ\ldots\circ\sigma_W=\tau|s_0\oplus\sigma_1\circ\ldots\circ\sigma_W=s),\mathcal{D})\\
\le&\frac{\epsilon}{|C|}\sum_{t\in C}\sum_{\tau}\Pr\{\sigma_{L+1}\circ\ldots\circ\sigma_W=\tau|t\oplus\sigma_{L+1}\circ\ldots\circ\sigma_W=s\}\\
=&\frac{\epsilon}{|C|}\cdot|C|=\epsilon.
\end{aligned}$$

For the second part, note that we have (similar to the above)

$$\begin{aligned}
&\Pr\{\sigma_{L+1}\circ\ldots\circ\sigma_R=\tau|s_0\oplus\sigma_1\circ\ldots\circ\sigma_W=s\}\\
=&\sum_{t,t'\in C}\Pr\{\sigma_{L+1}\circ\ldots\circ\sigma_R=\tau|t\oplus\tau=t',s_0\oplus\sigma_1\circ\ldots\circ\sigma_L=t,t'\oplus\sigma_{R+1}\circ\ldots\circ\sigma_W=s\}\cdot\\
&\qquad\Pr\{s_0\oplus\sigma_1\circ\ldots\circ\sigma_L=t,t'\oplus\sigma_{R+1}\circ\ldots\circ\sigma_W=s\}
\end{aligned}$$

Now, t is the last state of a random walk from s_0 and t' is the last state of a random walk from s. The latter random walk is the reverse of $\sigma_{R+1}\circ\ldots\circ\sigma_W$ with all edges reversed in C (denoted the edge-reversed component by C'). It is clear that C' is strongly connected. Let π' be the stationary distribution on C' under the transition induced by $\mathrm{Unif}(S_0^r)$, where S_0^r denotes the reverse streams of S_0. If $L\gg 1$ and $R\ll W$, both walks $\sigma_1\circ\ldots\circ\sigma_W$ and $\sigma_W^r\circ\ldots\circ\sigma_{R+1}^r$ will mix. By Markov property,

$$\begin{aligned}
&\Pr\{s_0\oplus\sigma_1\circ\ldots\circ\sigma_L=t,t'\oplus\sigma_{R+1}\circ\ldots\circ\sigma_W=s\}\\
=&\Pr\{t'\oplus\sigma_{R+1}\circ\ldots\circ\sigma_W=s|s_0\oplus\sigma_1\circ\ldots\circ\sigma_L=t\}\cdot\Pr\{s_0\oplus\sigma_1\circ\ldots\circ\sigma_L=t\}\\
=&\Pr\{t'\oplus\sigma_{R+1}\circ\ldots\circ\sigma_W=s\}\Pr\{s_0\oplus\sigma_1\circ\ldots\circ\sigma_L=t\}
\end{aligned}$$

which can be made close to $\pi(t)\pi'(t)$ with an additive error at most $\epsilon/|C|^2$ with choice of L and R uniformly over s_0 and s, as there are only finitely many distinct s_0's and s's. It

follows that

$$\begin{aligned}
&\left|\Pr\{\sigma_{L+1}\circ\ldots\circ\sigma_R=\tau|s_0\oplus\sigma_1\circ\ldots\circ\sigma_W=s\}-\frac{1}{|S_0|^{R-L}}\right|\\
\leq&\left|\sum_{t,t'\in C}\Pr\{\sigma_{L+1}\circ\ldots\circ\sigma_R=\tau|t\oplus\sigma_{L+1}\circ\ldots\circ\sigma_R=t'\}\pi(t)\pi'(t')-\frac{1}{|S_0|^{R-L}}\right|\\
&\qquad+\frac{\epsilon}{|C|^2}\sum_{t,t'\in C}\Pr\{\sigma_{L+1}\circ\ldots\circ\sigma_R=\tau|t\oplus\sigma_{L+1}\circ\ldots\circ\sigma_R=t'\}\\
=&\frac{\epsilon}{|C|^2}\sum_{t,t'\in C}\Pr\{\sigma_{L+1}\circ\ldots\circ\sigma_R=\tau|t\oplus\sigma_{L+1}\circ\ldots\circ\sigma_R=t'\},
\end{aligned}$$

where we use the fact that

$$\sum_{t,t'\in C}\Pr\{\sigma_{L+1}\circ\ldots\circ\sigma_R=\tau|t\oplus\sigma_{L+1}\circ\ldots\circ\sigma_R=t'\}\pi(t)\pi'(t')=\frac{1}{|S_0|^{R-L}}.$$

To see this, imagine that $\sigma_{L+1}\circ\ldots\circ\sigma_R$ is a part of a two-sided infinitely long zero-frequency sequence. Finally, similar to before,

$$d_{TV}(\mathcal{L}(\sigma_{L+1}\circ\ldots\circ\sigma_R|\mathcal{E}),\mathrm{Unif}(S_0^{R-L}))\leq\frac{\epsilon}{|C|^2}\cdot|C|^2=\epsilon.$$ ◀

Acknowledgements. Yuqing Ai and Wei Hu were supported in part by the National Basic Research Program of China Grant 2011CBA00300, 2011CBA00301, and the National Natural Science Foundation of China Grant 61361136003. Yi Li was supported by ONR grant N00014-15-1-2388 when he was at Harvard University, where his major participation in this work took place. David Woodruff was supported in part by the XDATA program of the Defence Advanced Research Projects Agency (DARPA), administered through Air Force Research Laboratory contract FA8750-12-C-0323.

References

1 Noga Alon, Yossi Matias, and Mario Szegedy. The Space Complexity of Approximating the Frequency Moments. *JCSS*, 58(1):137–147, 1999.

2 Sepehr Assadi, Sanjeev Khanna, Yang Li, and Grigory Yaroslavtsev. Maximum matchings in dynamic graph streams and the simultaneous communication model. In *SODA*, pages 1345–1364, 2016.

3 Christos Boutsidis, David P. Woodruff, and Peilin Zhong. Optimal principal component analysis in distributed and streaming models. In *STOC*, 2016.

4 Sumit Ganguly. Lower bounds on frequency estimation of data streams. In *Proceedings of the 3rd International Conference on Computer Science: theory and applications*, CSR'08, pages 204–215, 2008.

5 Piotr Indyk. Stable distributions, pseudorandom generators, embeddings, and data stream computation. *J. ACM*, 53(3):307–323, 2006.

6 Piotr Indyk. Sketching, streaming and sublinear-space algorithms, 2007. Graduate course notes available at `http://stellar.mit.edu/S/course/6/fa07/6.895/`.

7 Piotr Indyk, Eric Price, and David P. Woodruff. On the power of adaptivity in sparse recovery. In *FOCS*, pages 285–294, 2011.

8 Daniel M. Kane, Jelani Nelson, and David P. Woodruff. On the exact space complexity of sketching and streaming small norms. In *SODA*, pages 1161–1178, 2010.

9 Yi Li, Huy L. Nguyen, and David P. Woodruff. Turnstile streaming algorithms might as well be linear sketches. In *STOC*, pages 174–183, 2014.

10 S. Muthukrishnan. Data Streams: Algorithms and Applications. *Foundations and Trends in Theoretical Computer Science*, 1(2):117–236, 2005.

11 Ilan Newman. Private vs. common random bits in communication complexity. *Information Processing Letter*, pages 67–71, 1991.

12 Joachim von zur Gathen and Malte Sieveking. A bound on solutions of linear integer equalities and inequalities. In *Proceedings of the American Mathematical Society*, pages 155–158, 1978.

13 Omri Weinstein and David P. Woodruff. The simultaneous communication of disjointness with applications to data streams. In *ICALP*, pages 1082–1093, 2015.

14 David P. Woodruff. Low rank approximation lower bounds in row-update streams. In *NIPS*, pages 1781–1789, 2014.

15 Andrew Chi-Chih Yao. Some complexity questions related to distributive computing. In *STOC*, pages 209–213, 1979.

A The LNW Reduction

We show that the reduction from a general automaton to a path-reversible automaton, presented in [9, Theorem 5], actually gives us a path-independent automaton.

The reduction works as follows. Let $\mathcal{A}$ be the original automaton, $G'_{\mathcal{A}}$ be its zero-frequency graph, and $\oplus$ be its transition function. The states of the new automaton $\mathcal{B}$ are defined to be the terminal strongly connected components of $G'_{\mathcal{A}}$. (A strongly connected component is terminal if there is no arc from it to the rest of the graph.) For each strongly connected component v of $G'_{\mathcal{A}}$, let $rep(v)$ be a (fixed) arbitrary vertex in v, and $\alpha(v)$ be a (fixed) arbitrary terminal strongly connected component reachable from v. For each vertex u in $G'_{\mathcal{A}}$, let $com(u)$ be the strongly connected component it belongs to. Then the transition function $\oplus'$ of $\mathcal{B}$ is defined as

$$v \oplus' \pm e_i = \alpha(com(rep(v) \oplus \pm e_i)),$$

where v is a state of $\mathcal{B}$, i.e., a terminal strongly connected component of $G'_{\mathcal{A}}$.

It is shown in [9, Lemma 6] that $\mathcal{B}$ is path-reversible:

▶ **Lemma A.1** (Lemma 6 in [9]). *For any state u of $\mathcal{B}$ and any $i \in [n]$, we have $u \oplus' e_i \circ -e_i = u$.*

We show a stronger result below, which implies $\mathcal{B}$ is path-independent.

▶ **Lemma A.2.** *For any state u of $\mathcal{B}$ and any zero-frequency stream σ, we have $u \oplus' \sigma = u$.*

Proof. Let $\sigma = (\sigma_1, \ldots, \sigma_t)$ ($\sigma_i \in \Sigma$) and $v = u \oplus' \sigma$. Then there exist zero-frequency streams $\gamma^1, \gamma^2, \ldots, \gamma^t$ such that

$$rep(u) \oplus \sigma_1 \circ \gamma^1 \circ \sigma_2 \circ \gamma^2 \circ \ldots \circ \sigma_t \circ \gamma^t = rep(v).$$

Note that $\mathrm{freq}(\sigma_1 \circ \gamma^1 \circ \sigma_2 \circ \gamma^2 \circ \ldots \circ \sigma_t \circ \gamma^t) = \mathrm{freq}(\sigma_1 \circ \sigma_2 \circ \ldots \circ \sigma_t) = \mathrm{freq}\,\sigma = \vec{0}$. Since $rep(u)$ belongs to a terminal strongly connected component u, $rep(v)$ has to be in the same terminal strongly connected component. Hence $u = com(rep(u)) = com(rep(v)) = v$. ◀

B An Alternative Upper Bound on the Length of Shortest Zero-Frequency Paths

▶ **Lemma B.1.** *Let $s = |V|$ be the number of vertices in the transition graph $G_{\mathcal{A}}$. Then $L \leq e^{O((s+n)\log s)}$, where L is the maximum length of the shortest zero-frequency path between any two vertices connected by at least one zero-frequency path.*

Proof. Consider two vertices $o_1, o_2 \in V$ such that there exists a zero-frequency path from o_1 to o_2. We fix a tuple (p, C) satisfying the following condition: every vertex in $c_1, \ldots, c_t$ can be reached from o_1 via arcs in $c_1, \ldots, c_t$ and p. Here p is a simple path from o_1 to o_2 and $C = \{c_1, c_2, \ldots, c_t\}$ is a set of simple cycles in $G_{\mathcal{A}}$. For every possible (p, C) satisfying this condition, we build a linear system $Ax = b$ and want a positive integer solution. Here $A = \begin{pmatrix} f_{c_1} & f_{c_2} & \cdots & f_{c_t} \end{pmatrix}$ is an $n \times t$ matrix and $b = -f_p \in \mathbb{R}^n$. The i-th row in the equations guarantees the i-th component in the frequency of the path is 0. Each positive integer solution x of the equations corresponds to a zero-frequency path from o_1 to o_2 using the simple path p and simple cycles in C. The path p is passed through exactly once and each cycle c_i is passed through x_i times.

By Lemma 3.1, if there exists a positive integer solution, then there exists such a solution x so that $\|x\|_\infty \leq (t+1)^2 M_1$, where M_1 is the largest possible absolute value of any sub-determinant of $\begin{pmatrix} A & b \end{pmatrix}$. Since $c_1, \ldots, c_t$ and p are simple, they all have length at most s. Thus, the sum of the absolute values of entries in any column of $\begin{pmatrix} A & b \end{pmatrix}$ is bounded by s. By the Gershgorin circle theorem, the eigenvalues of any submatrix of $\begin{pmatrix} A & b \end{pmatrix}$ have absolute value at most s. Therefore, M_1 is at most s^n. On the other hand, for the number of simple cycles in a graph with s vertices, we have $t = |C| \leq s^{O(s)}$. Therefore

$$\|x\|_\infty \leq (s^{O(s)} + 1)^2 \cdot s^n = s^{O(n+s)}.$$

Thus, the length of the corresponding path is bounded by

$$s\|x\|_1 \leq st\|x\|_\infty \leq s \cdot s^{O(s)} \cdot s^{O(n+s)} = s^{O(n+s)}.$$

Since there exists a zero-frequency path from o_1 to o_2, we can decompose it into the sum of a simple path from o_1 to o_2 and a linear combination of simple cycles. This leads to the existence of a positive integer solution for the equations $Ax = b$ when we fix this path as p and this set of cycles as C. Therefore, there exists a zero-frequency path from o_1 to o_2 whose length is at most $s^{O(n+s)}$. ◀

Evolution and Computation

Nisheeth K. Vishnoi

École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland
nisheeth.vishnoi@epfl.ch

Abstract

Over the last two centuries there have been tremendous scientific and mathematical advances in our understanding of evolution, life and its mysteries. Recently, the relatively new and powerful tool of computation has joined forces to develop this understanding further: the underlying tenet is that several natural processes, including evolution itself, can be viewed as computing or optimizing something – *evolution is computation.* Furthermore, as in computation, efficiency is an important consideration in evolution. As many of these evolutionary processes are described using the language of dynamical systems, this entails understanding how quickly such systems can attain their equilibria. This endeavor not only has the potential to give us fundamental insights into life, it holds the promise that we will unveil new computational models and techniques. In this talk we will see some vignettes of this interplay between evolution and computation.

Keywords and phrases Evolution, Dynamical Systems, Algorithms, Complexity

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.21

Category Invited Talk

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 21; pp. 21:1–21:1

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

Tight SoS-Degree Bounds for Approximate Nash Equilibria

Aram W. Harrow[1], Anand V. Natarajan[2], and Xiaodi Wu[3]

1 **Center for Theoretical Physics, Massachusetts Institute of Technology, Cambridge, USA**
2 **Center for Theoretical Physics, Massachusetts Institute of Technology, Cambridge, USA**
3 **Department of Computer and Information Science, University of Oregon, Eugene, USA**

Abstract

Nash equilibria always exist, but are widely conjectured to require time to find that is exponential in the number of strategies, even for two-player games. By contrast, a simple quasi-polynomial time algorithm, due to Lipton, Markakis and Mehta (LMM), can find *approximate* Nash equilibria, in which no player can improve their utility by more than ϵ by changing their strategy. The LMM algorithm can also be used to find an approximate Nash equilibrium with near-maximal total welfare. Matching hardness results for this optimization problem were found assuming the hardness of the planted-clique problem (by Hazan and Krauthgamer) and assuming the Exponential Time Hypothesis (by Braverman, Ko and Weinstein).

In this paper we consider the application of the sum-squares (SoS) algorithm from convex optimization to the problem of optimizing over Nash equilibria. We show the first unconditional lower bounds on the number of levels of SoS needed to achieve a constant factor approximation to this problem. While it may seem that Nash equilibria do not naturally lend themselves to convex optimization, we also describe a simple LP (linear programming) hierarchy that can find an approximate Nash equilibrium in time comparable to that of the LMM algorithm, although neither algorithm is obviously a generalization of the other. This LP can be viewed as arising from the SoS algorithm at $\log n$ levels – matching our lower bounds. The lower bounds involve a modification of the Braverman-Ko-Weinstein embedding of CSPs into strategic games and techniques from sum-of-squares proof systems. The upper bound (i.e. analysis of the LP) uses information-theory techniques that have been recently applied to other linear- and semidefinite-programming hierarchies.

1998 ACM Subject Classification F.2 [Theory of Computation] Analysis of Algorithms and Problem Complexity

Keywords and phrases Approximate Nash Equilibrium, Sum of Squares, LP, SDP

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.22

1 Introduction

1.1 Motivation and background

Game theory, broadly speaking, seeks to explain the decision-making of interacting, self-interested agents. Mathematically it can be seen as a generalization of optimization problems in which the goal is to minimize or maximize some function. Instead each player wishes to maximize its own payoff, which in general will depend on the actions of the other players as well. The standard solution concept here is a Nash equilibrium, meaning a set

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 22; pp. 22:1–22:25

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

of (uncoordinated) mixed strategies for which no player can unilaterally increase their own payoff by changing strategy. Here "mixed strategy" means a probability distribution over the basic "pure strategies" of the game and "uncoordinated" means that these distributions are uncorrelated. A related notion is a "correlated equilibrium" where again players cannot unilaterally improve their payoffs, but this time a "signal" random variable is broadcast to all the players who are free to choose their strategy based on the common signal; e.g. consider the role of a traffic light in suggesting that one car stop and another car drive.

As with optimization problems, the practical applicability of Nash equilibria and correlated equilibria depend on their computational complexity. Nash's 1950 existence theorem proved that Nash equilibria exist under very general conditions [24, 25] but turning the proof into an algorithm results in an exponential runtime. Indeed, finding a Nash equilibrium is PPAD-complete, meaning that it is as hard as solving an abstract fixed-point problem [10]. If we instead consider the problem of maximizing a linear function (e.g. total payoff) over the space of Nash equilibria then the problem becomes NP-complete [10]. The difference in complexity (PPAD vs NP) reflects the fact that the former is a problem of searching for a solution that is known to exist whereas the latter problem is to determine whether a solution exists. Finding a correlated equilibrium, by contrast, can be achieved in poly time with linear programming.

One could reasonably argue that exact Nash equilibria are implausible models of rational behavior and that deviating from an equilibrium strategy might only happen in practice when the benefit is greater than zero by some non-negligible amount. An ϵ-approximate Nash equilibrium (aka. ϵ-ANE) is thus defined to be a set of uncorrelated strategies for which no player can improve their payoff by more than ϵ by changing strategies. It turns out that the complexity of finding ϵ-ANEs is significantly lower than that of exact Nash equilibria. In 2003, Lipton, Markakis and Mehta [21] gave an algorithm for finding an ϵ-ANE in quasipolynomial time, e.g. $n^{O(\log(n)/\epsilon^2)}$ for two-player games where both strategy sets have size n. Their algorithm was based on enumeration over a suitably chosen net of strategies. This net-based framework has been refined in [2, 6] to yield PTASs in some special cases. On the hardness side, Braverman, Ko and Weinstein [9] recently showed that finding the best (i.e. highest total payoff) ϵ-ANE in $n^{o(\log n)}$ time would violate the Exponential Time Hypothesis (ETH). (The ETH posits that 3-SAT instances on n variables require $\exp(\Omega(n))$ time.)

1.2 Main results

Our paper investigates ϵ-ANE from the perspective of convex optimization. Since the sets of Nash equilibria and ϵ-ANE are not convex, it is not immediately obvious how to relate the problem of finding an ANE to a convex optimization problem. To this end, we can consider the convex hull of all ϵ-ANE, for a given set of payoff functions. Optimizing a linear function over one of these sets is equivalent to optimizing a linear function over the set of ϵ-ANE. Note that these sets can be far from the much-more-tractable set of correlated equilibria; additionally, even though it is easy to test whether a strategy is an ϵ-NE, this does not extend to a test for whether a strategy is in the convex hull of ϵ-ANE (and likewise for Nash equilibria). Indeed, standard arguments mean that optimizing a linear function over or testing membership in these sets have approximately the same complexity [13].

Our first main result is a no-go theorem for a family of approximation algorithms based on semidefinite programming (SDP), called the sum of squares (SoS) hierarchy. In particular we show that in order to achieve a constant-factor approximation for 0.1-ANE, one must go to a level of at least $\Omega(\log n / \operatorname{poly}\log\log n)$ in the SoS hierarchy (where n is the size of the game).

This translates to an SDP of size $\Omega(n^{\log n})$. Unlike all previous results on the hardness of NE and ANE, our result is unconditional; i.e. does not depend on any assumptions about the hardness of 3-SAT, planted clique or other problems. Our result is also surprising in part because the best results for planted clique, an apparently comparable problem [16], only extend to ruling out SDPs arising from the SoS hierarchy of size $O(n^4)$ [17].

▶ **Theorem 1.1** (informal). *Given a game of size N with payoffs bounded by a constant, deciding whether either* (1) *there exists a Nash equilibrium with average payoff ≥ 1 or* (2) *all 0.1-approximate Nash equilibria have average payoff at most δ requires at least $\Omega(\log N/\operatorname{poly}\log\log N)$ levels in the SoS hierarchy.*

Our proof makes use of a classic result of Grigoriev [12], showing hardness for the problem 3XOR in the SoS model. We use reductions (with some properties as discussed below) to extend the hardness of 3XOR to ANE. There have been quite a few examples using reductions to prove the hardness in the SoS model (e.g., [34, 27]). However, each proof requires slightly different properties about reductions and there is no explicit unified framework for doing so. We follow a recent result in [15], which aims to serve as one such framework to facilitate the proof of hardness in the SoS model.

To obtain integrality gaps in the SoS model, one needs to show that (a) the SoS solution believes the value is large up to some high level (degree), and (b) the true value is actually small. To achieve (a), we follow the notion of *low-degree reductions* [15], in which one requires the reductions preserve a SoS solution for the reduced problems with almost the same value and a small amount of loss of the degree[1]. To achieve (b), we need the reductions to have some kind of *soundness*.

Our specific reduction is a variant of the one used in proving the ETH-hardness of ANE by Braverman, Ko and Weinstein [9], with tweaks to ensure it has *low-degree, soundness*, and *embedding* properties. Our hardness analysis is also inspired by a recent result of ours [15] that extends the SoS lower bounds to quantum information problems. This connection is natural given the intimate relationship between [1] and quantum information, which serves as the first step in the reduction of [9].

We also make use of the explicit construction of a two-player strategic game for which the optimal payoff of a Nash equilibrium is related to the value of a constraint satisfaction problem, in the second step of [9]. In their game, the two players each specify an assignment to a subset of $\sqrt{n}$ variables from the CSP, and receive a payoff if they are consistent with each other and satisfy the clauses. There are further penalties that ensure that each player must choose their subset close to uniformly at random. It is then proven that if the underlying CSP is satisfiable, the optimal Nash equilibrium is an "honest" strategy, where both players answer according to a fixed assignment to the variables. This establishes reductions with good completeness and soundness from CSPs to HonestNash of optimizing over honest strategies to this game. We prove that their reduction is also low-degree and pseudosolution-preserving, which allows us to obtain an SoS hardness result for HonestNash.

However, this game is not convenient for obtaining hardness for ApproximateNash, since not all honest strategies are in fact Nash equilibria. The problem is that Alice and Bob are punished for honest strategies that do not satisfy clauses. Additionally, the game depends on underlying CSP. We fix both problems at once by giving Alice and Bob a payoff

[1] This roughly refers to the "Vector Completeness" in [34] and "SoS Completeness" in [27]. It explicitly requires the existence of a mapping that is a polynomial of low-degree, which maps a SoS solution of the original problem to a SoS solution of the reduced one.

simply for answering consistently, independent of the clauses. We then offload all dependence on clause structure into our objective function.[2] Our objective function is 0 whenever Alice or Bob output sets of variables that do not contain enough clauses and otherwise equals the fraction of satisfied clauses. Maximizing this objective function over ϵ-ANE is then roughly equivalent to maximizing the number of satisfied clauses, which completes our reduction.

Our second main result is a linear programming (LP) lift of NE with dimension $n^{O(\log(n)/\epsilon^2)}$, matching our above lower bound. We describe lifts formally in Section 4, but intuitively an approximate lift of dimension D is a polytope in $\mathbb{R}^D$ whose projection onto some lower-dimensional space includes all Nash equilibria, but ideally not too many additional points. In Section 4 we prove the following theorem (more formally stated in Theorem 4.1).

▶ **Theorem 1.2** (informal). *Consider a two-player game with strategy sets of size n_1, n_2 and payoffs in $[-1, 1]$. There exists a polytope in $\mathbb{R}^D$ with $D = \exp(O(\log(n_1)\log(n_2)/\epsilon^2))$ such that its projection onto $\mathbb{R}^{n_1 n_2}$ contains all Nash equilibria and is contained in the ϵ-neighborhood of the convex hull of all ϵ-ANE. This polytope has an explicit efficient description, so that we can find an ϵ-ANE in time* $\mathrm{poly}(D)$*; or if f is an efficiently computable concave function, we can estimate its maximum value over the NE efficiently.*

The same result holds for m-player games where each player has a strategy set of size n if we set $D = \exp(O(m^3 \ln^2(n)/\epsilon^2))$.

In fact, the set of correlated equilibria can already be seen as an LP relaxation of Nash equilibria, since Nash equilibria can be alternately defined as correlated equilibria that are product distributions. This relaxation can be useful [28], but in general correlated equilibria can be far from Nash equilibria. Our lift can be thought of as a systematic hierarchy of successive refinements of the set of correlated equilibria. Inspired by [31, 7], our idea is to replace the player Bob with k replicas: Bob-1, Bob-2, ..., Bob-k. We will impose the constraint that the strategies of Alice and Bob-j form a correlated equilibrium, even when conditioned on the strategies of Bobs-$1, \ldots, j-1$. This approach prevents Alice from being simultaneously correlated with all of the Bobs. Indeed, if Alice were correlated with Bob-1, then conditioning on his strategy would reduce Alice's entropy. Continuing in this way we find that Alice must have low correlation with *most* of the Bobs, implying that if we choose j randomly from $\{1, \ldots, k\}$ and condition on the random strategies of Bobs-$1, \ldots, j-1$, then the resulting distribution on Alice and Bob-j will be nearly product. This means the resulting correlated equilibrium can be easily rounded to an ϵ-ANE.

The LP we obtain can be viewed as arising from the SoS hierarchy at level $\log N$, with the omission of the positive semidefinite constraint. Thus, our analysis of the LP also implies that SoS is able to solve approximate ApproximateNash to constant accuracy at level $\log N$, matching our SoS lower bound.

Our LP relaxation is not the first approximation algorithm for ANE. In fact, a nearly identical runtime was achieved by Lipton, Markakis and Mehta in 2003 [21] using an algorithm that exhaustively searched over a set of sparse strategies. Using Chernoff bounds it is possible to show that any NE can be approximated in this way. By now we have seen a series of examples where net-based algorithms and LP/SDP hierarchies give very similar approximation guarantees, often in the regime of PTASs or quasipolynomial-time algorithms. These examples are summarized in Table 1 of [8] and include (1) optimizing polynomials over

[2] This has the advantage of making our results compatible with the framework of extension complexity, where one considers a polytope of feasible solutions (e.g. the matching polytope) that is independent of the function being maximized.

the simplex, (2) optimizing polynomials over the unit sphere, (3) free two-player games, (4) unique games, (5) small-set expansion and (6) optimizing linear functions over unentangled quantum states (see [8] for specific references). Despite this series of coincidences, it is still an open question to find a common explanation of the performance of both types of algorithms. Indeed our algorithm differs from that of LMM in ways that suggest there is no obvious way to map one onto the other. If there is a sparse NE then LMM will find it, while our algorithm may not. On the other hand, while both algorithms can freely add linear constraints (e.g. on the total payoff), only ours can add convex constraints, such as maximizing entropy. Indeed, if there exists an NE with entropy $\geq c \log(n)$ then our algorithm will always find a nearby ϵ-ANE with entropy $\geq (c - O(\epsilon)) \log(n)$, while by construction, LMM will only find ϵ-ANE with entropy $\leq \log\log(n/\epsilon) + O(1)$. We further compare the algorithms in Section 5.

1.3 Open problems

- *Extension complexity.* Our results show limitations on approximating Nash equilibria using the SoS hierarchy. But what about more general SDPs? Is it possible to find an $n^{\Omega(\log n)}$ lower bound on the extension complexity of ϵ-ANE? The approach of Lee, Raghavendra, and Steurer [20] does not apply directly here because our problems do not have the same self-embedding property that CSPs do. However, it seems likely that the [20] framework can be extended to cover ϵ-ANE.
- *Special cases.* While our results address the complexity of ϵ-ANE in the worst case, there are many special cases where it should be possible to find more efficient convex relaxations. Under various conditions on the payoff matrices, net-based algorithms can run more quickly [2, 6]. Without any further modification, our algorithm is already more effective when Alice has low entropy in all correlated equilibria. This condition can be checked quickly (since correlated equilibria form a polytope and entropy is a concave function) but appears incomparable to the conditions under which [21, 2, 6] outperform their worst-case guarantees. More generally we would like to know scenarios under which our algorithms or variants of them can perform significantly better.
- *Semidefinite and convex constraints.* A related question is whether SDP or other convex constraints give additional benefits not already captured by LP hierarchies. We could also add concave objective functions, such as entropy maximization. Do these have further application?
- *Search vs. optimization.* A major theme in work on the complexity of finding Nash equilibrium is the distinction between NP and PPAD. PPAD is an example of TFNP ("total function NP") which is the class of search problems for which an answer is guaranteed to exist and can be efficiently verified. All known algorithms for finding Nash equilibria can also perform (or approximate) the optimization versions of the problem, and the extension complexity model (by contrast with earlier hardness-based lower bounds) collapses the difference between the search and optimization versions. Is there a natural computational model which separates the complexity of these tasks?
- *Densest subgraph.* Techniques for proving upper and lower bounds on the complexity of the approximate Nash equilibrium problem have also been applied to the problem of finding the densest k-subgraph of a given graph. The lower bound of [16] rules out a PTAS for additive approximations to this problem on bipartite graphs, while the algorithm of [6] achieves a quasipolynomial algorithm. Can we match these results in the SoS or extension complexity settings? One barrier is that the hardness of [16] uses a reduction from the planted clique problem, for which SoS lower bounds are not as well understood as they are for CSPs.

1.4 Overview

In Section 2, we introduce some basic definitions and give a more technical overview of our contribution. In Section 3, we prove a lower bound on SoS relaxations for the set of two-player approximate Nash equilibria. Following this, in Section 4, we introduce an LP relaxation for approximate Nash that matches our lower bound as well as handling multiplayer games, and in Section 5, we compare its performance to the LMM algorithm. The appendices contain further background on SoS proofs (Appendix A), reductions for SDPs (Appendix B) and information theory (Appendix C).

2 Definitions and Preliminaries

2.1 Games

Consider a two-player game with strategy sets $[n_1], [n_2]$ and payoff vectors $f_1, f_2 \in \mathbb{R}^{n_1 n_2}$. A Nash equilibrium is a pair of probability distributions $p_1 \in \Delta_{n_1}, p_2 \in \Delta_{n_2}$ such that

$$\langle e_x \otimes p_2, f_1 \rangle \leq \langle p_1 \otimes p_2, f_1 \rangle \qquad \forall x \in [n_1] \tag{2.1a}$$

$$\langle p_1 \otimes e_y, f_2 \rangle \leq \langle p_1 \otimes p_2, f_2 \rangle \qquad \forall y \in [n_2] \tag{2.1b}$$

Here $[n] = \{1, \ldots, n\}$, $\Delta_n = \{p \in \mathbb{R}^n : p(x) \geq 0, \sum_x p(x) = 1\}$, e_x is the vector with a one in position x and zeroes elsewhere and $p \otimes q$ is the vector with x, y entry equal to $p(x)q(y)$.

Nash proved that (2.1) always has a solution (known as a Nash equilibrium, or NE), but finding one is known to be PPAD-complete (or NP-complete in some cases when additional constraints or optimizations are added) [10] . For this reason, it is natural to consider instead approximate NE. Assume for the rest of the paper that $\max_i \max_x |f_i(x)| \leq 1$. We say that the distributions p_1, p_2 (or equivalently the joint distribution $p_1 \otimes p_2$) are an ϵ-approximate NE (or ϵ-ANE) if they satisfy

$$\langle e_x \otimes p_2, f_1 \rangle \leq \langle p_1 \otimes p_2, f_1 \rangle + \epsilon \qquad \forall x \in [n_1] \tag{2.2a}$$

$$\langle p_1 \otimes e_y, f_2 \rangle \leq \langle p_1 \otimes p_2, f_2 \rangle + \epsilon \qquad \forall y \in [n_2] \tag{2.2b}$$

From these expressions, we can see that the problem of optimizing over Nash equilibria is a polynomial optimization problem, where the variables are the probabilities p and q. The constraints are the simplex constraints and the Nash conditions ((2.2)).

We consider also *correlated equilibria*, first proposed by Aumann in 1974 [3]. Let q^{XY} denote a probability distribution in $\Delta_{n_1 n_2}$. Then we say that q is a *correlated equilibrium* if q satisfies the following analogue of (2.1):

$$\sum_{y \in [n_2]} q(x, y)(f_1(x', y) - f_1(x, y)) \leq 0 \qquad \forall x, x' \in [n_1] \tag{2.3a}$$

$$\sum_{x \in [n_1]} q(x, y)(f_2(x, y') - f_2(x, y)) \leq 0 \qquad \forall y, y' \in [n_2] \tag{2.3b}$$

Since (2.3) is an LP, we can find correlated equilibria efficiently; i.e. in time $\text{poly}(n_1, n_2)$.

While our hardness results will focus on the two-player case, we also describe algorithms for finding ϵ-ANE for games with more than two players. Consider an m-player game, where the players have strategy sets $S_1 = [n_1], \ldots, S_m = [n_m]$ (with $S = S_1 \times \cdots \times S_m$), and payoff tensors $f_1, \ldots, f_m \in \mathbb{R}^{n_1} \otimes \cdots \otimes \mathbb{R}^{n_m}$. This means that if players use mixed strategies $p_1 \in \Delta_{n_1}, \ldots, p_m \in \Delta_{n_m}$, then player i receives payoff

$$\langle p_1 \otimes \cdots \otimes p_m, f_i \rangle = \sum_{x=(x_1,\ldots,x_m) \in S} p_1(x_1) \cdots p_m(x_m) f_i(x).$$

Let $N := \prod_{i=1}^{m} n_i$. A distribution $p \in \Delta_N$ is a correlated equilibrium if

$$\sum_{x_{-i} \in S_{-i}} p(x_i, x_{-i})(f_i(x_i', x_{-i}) - f_i(x_i, x_{-i})) \leq 0 \qquad \forall i \in [m], \forall x_i, x_i' \in S_i. \tag{2.4}$$

Here the notation S_{-i} indicates the strategy set $S_1 \times \cdots \times S_{i-1} \times S_{i+1} \times \cdots S_m$ of all players except player i, and similarly $x_{-i} := (x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_m)$. A Nash equilibrium is a correlated equilibrium that is also a product distribution: i.e. such that $p = p^{X_1} \otimes \cdots \otimes p^{X_m}$.

2.2 Norms

Define the 1-norm and ∞-norm of vectors to be

$$\|v\|_1 := \sum_x |v(x)| \qquad \text{and} \qquad \|v\|_\infty = \max_x |v(x)|.$$

For two probability distributions p, q, the 1-norm distance $\|p-q\|_1$ is also called the *variational distance*, because of the following special case of Hölder's inequality:

$$\langle v, w \rangle \leq \|v\|_1 \|w\|_\infty \qquad \text{with equality holding iff } v = \lambda w \text{ for } \lambda \geq 0. \tag{2.5}$$

2.3 Optimization

Both our upper and lower bounds apply to relaxations of the convex optimization problem ApproximateNash$_\epsilon$. The promise version of this problem is as follows:

▶ **Definition 2.1.** The problem $(a, b) -$ ApproximateNash$_{\epsilon,m}(f)$ is to determine, given a game G over m players, where each player has at most n deterministic strategies, and a function h (called the "externality") mapping strategies to real numbers, whether either

- there exists an exact Nash equilibrium strategy for G for which $h \geq a$, or
- for every ϵ-approximate Nash equilibrium to G, h is at most b.

given the promise that one of these cases holds. Here $b < a$ are constant parameters. As an important special case, we write ApproximateNash$_\epsilon$ to refer to the case when $m = 2$.

3 SoS Lower Bound

Braverman et al. [9] show a reduction from 3SAT to approximate Nash equilibrium, which shows hardness for this problem conditional on ETH. We are able to use their reduction to show an unconditional hardness result for the SoS hierarchy for approximate Nash. We follow the approach of [15], who show how to SoS hardness for several continuous-variable optimization problems arising in quantum information. The proof consists of two steps: a *pseudo-solution-preserving* reduction from 3XOR over n variables to the intermediate optimization problem HonestNash over $\{\pm 1\}^n$, followed by a reduction from HonestNash to approximate ApproximateNash. The following schematic diagram illustrates this:

$$\text{3XOR} \implies \text{HonestNash} \implies \text{ApproximateNash}.$$

- The problem HonestNash is a polynomial optimization problem over the boolean hypercube $\{\pm 1\}^n$. The objective function $h_\phi(x)$ is the total expected externality of two players in a particular strategic game, whose strategies are specified by the input variables x; the externality function is induced by a CSP instance ϕ. We consider the strategic game introduced by [9], for which this objective function is a polynomial of degree $\tilde{O}(\sqrt{n})$. This game, in turn, is based on a free game introduced by [1].

3.1 Framework of Deriving SoS Lower Bounds

Our proof makes extensive use of reductions between optimization problems. Here, we will briefly give definitions of some useful notions, while we defer a full description to Appendix B. Throughout this section, we will use the so-called ± 1 notation for boolean variables: that is, we encode FALSE as 1 and TRUE as -1.

We derive all of our integrality gaps from the following foundational result of Grigoriev for the problem 3XOR (defined in Section B.3):

▶ **Proposition 3.1** (Theorem 3.1 of [5], due to Grigoriev)**.** *For any $\epsilon > 0$, for every n there exists a* 3XOR *instance Φ_n with n variables and $m = O(n/\epsilon^2)$ clauses, such that $\mathrm{OPT}(\Phi_n) \leq \frac{1}{2} + \epsilon$, but there exists a degree-$\Omega(n)$ value-1 pseudo-solution $\tilde{\mathbb{E}}$.*

Here "value 1" means that for every clause $x_i x_j x_k = a_{ijk}$, it holds that $\tilde{\mathbb{E}}[(x_i x_j x_k - a_{ijk})p(x)] = 0$ for all polynomials $p(x)$ with degree at most $d - 3$.

The instance of Φ_n produced by Grigoriev has a constraint graph which is a good expander. However, there is no upper bound on the degree of the constraint graph, i.e. the number of clauses each variable can participate in. We remedy this issue by transforming the instance to an instance Φ'_n of the problem 3XOR+EQ, where we allow both 3XOR constraints and equality constraints between pairs of variables.

▶ **Proposition 3.2.** *For every n, there exists an instance Φ'_n of* 3XOR+EQ *on $O(n)$ variables where the constraint graph is $(\delta n, \alpha)$-expanding and has degree at most d, for constants $\delta < 1, \alpha > 1, d$, such that the maximum fraction of clauses satisfiable is $\omega(\Phi'_n) \leq \frac{1}{2} + \epsilon$. Furthermore, there exists a degree-$\Omega(n)$, value-1 pseudo-solution to Φ'_n. That is, there exists a pseudo-expectation operator $\tilde{\mathbb{E}}[\cdot]$ with degree $D = \Omega(n)$, such that $\tilde{\mathbb{E}}[C(x)q(x)] = 1$ for every clause $C(x)$ in the instance Φ'_n and every polynomial $q(x)$ with degree $\deg(C(x)q(x)) \leq D$.*

Proof. We start with the instance Φ_n in Proposition 3.1, and then apply the degree reduction procedure of [29] to produce the new instance Φ'_n. This procedure consists of replacing each high-degree variable in the original instance with many copies, connected by equality constraints laid out according to an expander graph. It is shown in [29] that this procedure has constant soundness, so $\omega(\Phi_n) \leq \frac{1}{2} + \epsilon$ for some constant ϵ. Now, to produce the desired pseudosolution, we define $\tilde{\mathbb{E}}[p(x)]$ for any polynomial $p(x)$ to be what we get by first *identifying* all of the replicated variables with each other, and then evaluating the pseudoexpectation according to the operator $\tilde{\mathbb{E}}[\cdot]$ produced by Proposition 3.1. ◀

3.2 Consistent Sample Game

Inspired by the construction in [9], we now present a game called the Consistent Sample Game, whose Nash equilibria will be easy to characterize. As a warm-up, consider the following simple game

▶ **Definition 3.3** (Consistent Bit Game)**.** The consistent bit game is a two-player strategic game, where Alice and Bob are each allowed to play a single bit 0 or 1. They win if they their bits agree and lose otherwise.

It is easy to see that this game has exactly two Nash equilibria: either Alice and Bob both play 0, or both play 1. Our consistent sample game is a scaled up version of this game, where Alice and Bob have access to n-bit strings. However, the strategies they play consist of assignments to a subset of the variables of size $\sqrt{n}$. To force the players to choose their subsets with close to uniform probability, we also add a zero-sum uniformity test. This test

and its analysis are one of the main technical contributions of [9], and fortunately we will be able to mostly reuse their analysis without change.

▶ **Definition 3.4** (Consistent Sample Game)**.** For a given size n, the consistent sample game $G_{n,k,\ell,d}$ is specified by the following:

- The set of Alice's possible pure strategies consists of: all tuples (S, s) consisting of a subset S of k variables and an assignment s to these variables, and all subsets Y of $\rho\sqrt{n}$ variables. We refer to the former as "tuple strategies" and the latter as "subset strategies."
- The set of Bob's possible pure strategies consist of: all tuples (T, t) consisting of a subset T of ℓ variables and an assignment t to the variables in T, and all subsets Z of $\rho\sqrt{n}$ clauses. We likewise refer to these two types of strategies as "tuple strategies" and "subset strategies."
- The payoff matrix has a block structure. If both players play with tuple strategies (S, s) and (T, t), the payoff for Alice is 1, and the payoff for Bob is

$$f'(S,T,s,t) = \begin{cases} \beta_T f(S,T,s,t) & \text{if } |S \cap \mathcal{N}(T)| > \frac{d}{10(\epsilon^*)^2} \\ 0 & \text{otherwise} \end{cases},$$

  where the notation $\mathcal{N}(T)$ denotes the neighbors of the variables in T in the constraint graph, i.e. the set of clauses involving variables in T. Here, $f(S, T, s, t)$ is 1 if there are no inconsistencies in Alice and Bob's assignments to the variables and 0 otherwise, and $\beta_T \equiv \frac{1}{\Pr[|S\cap\mathcal{N}(T)|>\frac{d}{10(\epsilon^*)^2}]}$.
- If Alice plays with a tuple strategy (S, s) and Bob plays with a subset strategy Z, then if $S \cap Z \neq \emptyset$, Bob receives a payoff of K and Alice receives $-K$. Likewise, if Bob plays with a tuple strategy (T, t) and Alice plays with a subset strategy Y, then if $T \cap Y \neq \emptyset$, Alice receives a payoff of K and Bob receives $-K$.
- If both players play with a subset strategy, they both receive a payoff of 0.

In the above, $K > 1$ and $\epsilon^* < \frac{1}{2}$ are constant parameters, and $\rho = (\epsilon^*)/(c_2 \cdot K)$ where c_2 is an appropriately chosen constant.

As in Braverman et al., we choose k and ℓ to be $\Theta(\sqrt{n})$. The parameter d is equal to the degree of the constraint graph of the CSP ϕ, which we will use later to construct our externality function. Henceforth, we will denote the game simply by G_n. An important difference between our game and the one in Braverman et al. is that in our construction, when both players use tuple strategies, Alice always receives a payoff of 1.

We now define a Boolean optimization problem HonestNash by considering a restricted subset of strategies which we call "honest strategies." We assume a fixed CSP instance ϕ that is known to both players.

▶ **Definition 3.5.** For every $x \in \{\pm 1\}^n$, we define the *honest strategy according to x for G_n* by the following prescription

- Alice follows a mixed strategy: she chooses her subset of variables S to be $k/3$ clauses chosen uniformly at random from ϕ, and her assignment s to be that given by x.
- Bob also follows a mixed strategy: he chooses his subset T of ℓ variables uniformly at random from ϕ, and his assignment t to be that given by x.

▶ **Definition 3.6.** For every 3XOR instance ϕ, the problem HonestNash_ϕ is a Boolean optimization problem $\max_{x\in\{\pm1\}^n} h_\phi(x)$, where the objective function $h_\phi(x)$ is the expected value of the following externality function over the honest strategy induced by x (note that in an honest strategy, Alice and Bob always play tuple strategies):

- The externality is 1 if both players' assignments are consistent with each other and all of Alice's assignments satisfy their respective clauses in ϕ.
- The externality is 0 otherwise.

This objective function is a degree-$O(\sqrt{n})$ polynomial in the variables x.

To see why the function h_ϕ is a degree $O(\sqrt{n})$ polynomial, note that h can be written as an average of terms, where each term corresponds to the externality for a specific choice of S and T. This is a boolean function depending on only the $O(\sqrt{n})$ variables that appear in S and T. So the whole function $h_\phi(x)$ can be written as a sum of terms for each S, T, each of which has degree $O(\sqrt{n})$. Below in the proof of theorem 3.8, we will give an explicit expression for $h_\phi(x)$.

▶ **Theorem 3.7.** *If ϕ is satisfiable, then there exists an honest strategy for G_n that is an exact Nash equilibrium, and achieves expected externality* 1*. Moreover, there are fixed constants $\delta, \epsilon^* < 1/2$ independent of ϵ, such that if at most $(1-\delta)$-fraction of the clauses of ϕ are satisfiable, then all ϵ^*-approximate Nash equilibria for G_n have expected externality at most $O(\epsilon)$.*

Proof. This is the main result of [9]; we need to argue that is preserved under our modification of the payoff function. Because of the similarity between our proof and theirs, we only sketch our proof. For the completeness case, the desired Nash equilibrium is simply the honest strategy playing according to the satisfying assignment of ϕ. The argument presented in Lemma 3.2 of [9] goes through without change.

For the soundness, we again follow the proof strategy of [9]. In particular, we note that the proof of Lemma 3.4, which states that all Nash equilibria must choose the subsets S, T with roughly uniform probability over the clauses in ϕ, holds unchanged with our payoff function. Thus, we can reproduce the argument of Lemma 3.5 with our modified payoff function, to upper bound Bob's externality by $O(\epsilon)$. Moreover, the same argument applied to our payoff function upper-bounds Alice's externality by $O(\epsilon)$. Thus, we obtain an average payoff of $O(\epsilon)$ as desired. ◀

▶ **Theorem 3.8.** *For ϕ be the* 3XOR+EQ *instance from Proposition 3.2, there exists a degree-$\Omega(n)$ pseudosolution to* HonestNash *with externality h_ϕ achieving value* 1*.*

Proof. Let $\tilde{\mathbb{E}}[\cdot]$ be the degree-$\Omega(n)$ pseudoexpectation operator associated with a value-1 pseudosolution for our 3XOR+EQ instance ϕ. (Such a pseudosolution exists by proposition 3.2.) We claim that this yields a value-1 pseudosolution for HonestNash. To prove this, let us examine the objective function $f_\phi(x)$ for this problem. First, define the polynomial

$$\mathrm{AND}(x_1, x_2, \ldots, x_k) = 1 + \frac{1}{2^{k-1}}(1-x_1)(1-x_2)\ldots(1-x_k)$$

This polynomial evaluates to -1 when all of the input variables are -1, and 1 otherwise. Next, we define a polynomial function for $g_C(x)$ for each clause $C = ax_ix_jx_k$:

$$g_C(x) = 1 - \frac{1}{2}(x_ix_jx_k - a)^2$$

This evaluates to -1 when the clause is satisfied and 1 otherwise. By our definition of honest strategies, the consistency tests always pass with probability 1. So the objective function

$f_\phi(x)$ is a function of just the clauses $\{C_1, \cdots, C_k\}$ appearing in the random sets of variables S and T:

$$f_\phi(x) = \frac{1}{2} - \frac{1}{2} \mathop{\mathbb{E}}_S \mathop{\mathbb{E}}_T \beta_T \, \mathrm{AND}(g_{C_1}(x), g_{C_2}(x), \ldots, g_{C_k}(x)) = \frac{1}{2} - \frac{1}{2^k} \mathop{\mathbb{E}}_S \prod_{i=1}^{k} (1 - C_i(x)).$$

Here the expectations are taken over the uniform distribution over sets S, T of the appropriate size. Note that $k = \Theta(\sqrt{n})$. Now, we know that $\tilde{\mathbb{E}}'[C_i(x)q(x)] = 0$ for all $q(x)$ such that $\deg(q(x)) \leq d$ where $d = \Omega(n)$. Therefore,

$$\tilde{\mathbb{E}}'[\mathrm{AND}(g_{C_1}(x), g_{C_2}(x), \ldots, g_{C_k}(x))] = 1.$$

So $\tilde{\mathbb{E}}'[f_\phi(x)] = \mathbb{E}_S \, \mathbb{E}_T \, \beta_T = 1$. ◀

3.3 Embedding HonestNash in ApproximateNash

We now pass from HonestNash to ApproximateNash by broadening the space of strategies searched over to include *all* mixed strategies, not just honest ones. In order to preserve our SoS lower bound, it will help to show that every feasible point of HonestNash corresponds to a feasible point of ApproximateNash achieving exactly the same value.

▶ **Theorem 3.9.** *Every honest strategy for G_n is a Nash equilibrium.*

Proof. First, we will show that there is no incentive for either player to switch to another tuple strategy. We will then invoke the soundness analysis of Braverman et al. [9] to show that there is no incentive to switch to subset strategies either.

For tuple strategies, there are two cases the consider. First, let's suppose we fix Alice's strategy and allow Bob's strategy to deviate. Bob's payoff depends only on whether Alice's clauses are satisfied, and whether Alice and Bob are inconsistent on any variables. If Bob were to deviate from the honest strategy, the number of satisfied clauses would be unaffected, and the chance of inconsistencies can only go up. So Bob has no incentive to deviate. Now, if we fix Bob's strategy and allow Alice to deviate, note that Alice's payoff is always 1 regardless of which strategy she chooses, so she has no incentive to deviate either.

Now, we need to show that neither party has an incentive to switch to a subset strategy. We will use the following fact, which is shown in the course of the proof of Lemma 3.2 of [9].

▶ **Fact 3.10.** *Suppose Alice plays honestly and Bob plays with a (deterministic) subset strategy Z. Then Bob's expected payoff is upper bounded by*

$$v = K \mathop{\mathbb{E}}_{S \sim \mathcal{U}} \mathbf{1}(S \cap Z \neq \emptyset) \leq \frac{2}{0.9 \cdot c_2}.$$

So for $c_2 > 2/(0.9\epsilon^*)$, Bob has no incentive to deviate. A symmetric argument applies to Alice. ◀

We note that the previous theorem would be false in the original version of the game given by [9], without the modification to Alice's payoff. This is because for any honest assignment to the variables with 3XOR value at least $\frac{1}{2} + \epsilon$, Alice can find some subset S' of $\sqrt{n}$ clauses that are perfectly satisfied by that assignment. So Alice will always have an incentive to switch to the deterministic strategy that always answers with S': this strategy would achieve a payoff of 1 for Alice. We are able to remove this incentive by making Alice's payoff 1 independent of her choice of strategy.

It turns out that the preceding argument to show that each honest strategy is a Nash equilibrium can be itself converted into a sum of squares proof. This enables us to "lift" the SoS pseudosolution we constructed for HonestNash to one for ApproximateNash, and achieve our main lower bound.

▶ **Theorem 3.11.** *There exists a constant $\epsilon^* < \frac{1}{2}$ such that for any constant $\epsilon < \frac{1}{2}$, there exists a game G_n of size $N = O(n^{\sqrt{n}})$ and externality function h such that the expected value of h is at most ϵ over all ϵ^*-Nash equilibria. At the same time, there is a degree-$\Omega(\log N)$ pseudosolution that satisfies all the Nash equilibrium constraints for G_n and achieves externality value* 1.

Proof. Choose G_n to be the consistent sample game, and take the externality function h to be the one induced by the 3XOR+EQ instance from Proposition 3.2. Then it follows from Theorem 3.7 that the expected value of h is at most ϵ over all ϵ^*-Nash equilibria. Now, to construct the pseudosolution, first, let us define a polynomial formulation of the problem ApproximateNash. Our variables will be $p_{(S,s)}$, representing the probability that Alice plays the tuple strategy (S, s); p_Y, representing the probability that Alice plays the subset strategy Y; $q_{(T,t)}$ representing the probability that Bob plays the tuple strategy (T, t), and q_Z representing the probability that Bob plays the subset strategy Z. Let $\tilde{\mathbb{E}}[\cdot]$ be the pseudosolution for HonestNash derived in Theorem 3.8, which is defined up to degree $D = \Omega(\sqrt{n})$. We need to lift this to a pseudoexpectation $\tilde{\mathbb{E}}'[\cdot]$ on variables p, q. We do so as follows: when evaluating a pseudoexpectation $\tilde{\mathbb{E}}'[p \dots pq \dots q]$ of a monomial term, first perform the following substitutions, and then evaluate the resulting polynomial in x according to $\tilde{\mathbb{E}}[\cdot]$:

- Replace p_Y or q_Z with 0 (this is because honest strategies have no support over subset strategies).
- Replace $p_{(S,s)}$ by $c_a \,\mathrm{AND}(x_{S_1} = s_1, \dots, x_{S_k} = s_k)$, where c_a is the probability of Alice choosing the subset S. This term evaluates to c_a if the assignment in s matches the assignment in x, and 0 otherwise.
- Likewise, replace $q_{(T,t)}$ by $c_b \,\mathrm{AND}(x_{T_1} = t_1, \dots, x_{T_k} = t_k)$, where c_b is the probability of Bob choosing the subset T.

This pseudoexpectation "automatically" satisfies the positive semidefinite constraint, since it arises from a valid pseudoexpectation for x. Moreover, under the substitution process, the degree of a polynomial can only increase by a multiplicative fact of at most $O(\sqrt{n})$, so $\tilde{\mathbb{E}}'[\cdot]$ is defined up to degree $\Omega(D/\sqrt{n}) = \Omega(\sqrt{n})$. It remains to check that it satisfies the Nash constraints, and gives a high objective value for the externality function. To check the former, we need to show that the pseudoexpectation of the advantage gained by switching to any other strategy, multiplied by any squared polynomial, is non-positive. First, let us consider Bob. Suppose Bob switches to a tuple strategy (T', t'). The advantage gained can be written as

$$\text{advantage} = \sum_{(S,s),(T,t)} p_{(S,s)} q_{(T,t)} (f_{(S,s),(T',t')} - f_{(S,s),(T,t)}). \tag{3.1}$$

The term $f_{(S,s),(T,t)}$ checks whether the assignments s and t are consistent, so it is a degree-$O(\sqrt{n})$ polynomial in the variables x (essentially the AND of a number of equality checks). We want to check that

$$\tilde{\mathbb{E}}'[\text{advantage} \cdot P^2(p, q)] \leq 0$$

for an arbitrary polynomial $P(p, q)$. By the construction of the honest strategies, all (S, s, T, t) in the support of the strategy are consistent and so $\tilde{\mathbb{E}}'[(f_{(S,s),(T,t)} - 1)Q(p, q)] = 0$ for any

such (S, s, T, t) and any polynomial $Q(p, q)$. Another way to see this is that any *inconsistent* tuples (S, s, T, t) should disagree on some bit x_i, meaning that one AND term contains a $1 + x_i$ and the other AND term contains a $1 - x_i$. In this case

$$\tilde{\mathbb{E}}'[p_{(S,s)}q_{(T,t)}Q(p,q)] = \tilde{\mathbb{E}}[(1-x_i)(1+x_i)Q'(x)] = 0$$

where $Q'(x)$ is some other polynomial of $x_1, \ldots, x_n$ and the last equality comes from the $x_i^2 = 1$ constraint satisfied by the original pseudoexpectation $\tilde{\mathbb{E}}$. Thus the second term in (3.1) always evaluates to $\tilde{\mathbb{E}}'[P^2]$ under the pseudoexpectation. On the other hand we claim the first term is $\leq \tilde{\mathbb{E}}'[P^2]$. Here the constraint $\mathrm{AND}^2 = 1$ is implied by the $x_i^2 = 1$ constraints, thus we have the SOS proof

$$1 - \mathrm{AND} = \frac{1 - 2\,\mathrm{AND} + \mathrm{AND}^2}{2} = \frac{(1-\mathrm{AND})^2}{2} \geq 0.$$

This implies that $\tilde{\mathbb{E}}[\mathrm{AND}^2\, P^2] \leq \tilde{\mathbb{E}}[P^2]$, as desired.

Next, suppose Bob switches to a subset strategy Z. Then his advantage is

$$\text{advantage} = \sum_{(S,s),(T,t)} p_{(S,s)}q_{(T,t)}(f_{(S,s),Z} - f_{(S,s),(T,t)}). \tag{3.2}$$

Note that the first term in the difference is independent of s. Indeed the second term is as well, since $f_{(S,s),(T,t)} = 1$ whenever $p_{(S,s)}q_{(T,t)}$ are not constrained to equal 0 by our construction. Thus we can rewrite the advantage as

$$\text{advantage} = \sum_{S,T} p_S q_T (K\mathbf{1}(S \cap Z \neq \emptyset) - 1), \tag{3.3}$$

with $p_S := \sum_s p_{(S,s)}$ and likewise for q_T. We now claim that these terms factor out of a pseudoexpectation; i.e. that

$$\tilde{\mathbb{E}}'[p_S P(p,q)] = \tilde{\mathbb{E}}'[p_S]\,\tilde{\mathbb{E}}'[P(p,q)] \tag{3.4}$$

for any $P(p, q)$ (and likewise for q_T). To see this observe that our substitution replaces p_S with

$$\sum_{s_1,\ldots,s_k} c_a(1 + \frac{1}{2^{k-1}}(1 - s_1x_1)(1 - s_2x_2)\ldots(1 - s_kx_k)) = 2^k c_a.$$

Thus for any polynomial $P(p, q)$ we have

$$\tilde{\mathbb{E}}'[\text{advantage} \cdot P^2(p,q)] = \tilde{\mathbb{E}}'[\text{advantage}]\,\tilde{\mathbb{E}}'[P^2(p,q)]. \tag{3.5}$$

The first term is nonpositive due to Fact 3.10 and the second term is nonnegative, as we have argued above, because it equals $\tilde{\mathbb{E}}[\tilde{P}^2(x)]$ for some polynomial $\tilde{P}$ and because $\tilde{\mathbb{E}}$ is a pseudoexpectation.

We conclude that $\tilde{\mathbb{E}}'$ satisfies the Nash equilibrium constraints. It achieves externality value 1 because it inherits the property from $\tilde{\mathbb{E}}$ of satisfying all the 3XOR constraints. ◀

4 Approximate Nash equilibria via linear programming

In this section we will describe an LP hierarchy for ANE.

We first introduce a new form of equilibrium, called an ϵ-correlated equilibrium. If p is a correlated equilibrium for some m-player game f then we say it is an ϵ-correlated equilibrium if

$$\|p^{X_1 \dots X_m} - p^{X_1} \otimes \cdots \otimes p^{X_m}\|_1 \leq \epsilon. \tag{4.1}$$

Denote the set of Nash equilibria for game f by $\mathcal{N}_f$, the ϵ-correlated Nash equilibria by $\tilde{\mathcal{N}}_{f,\epsilon}$ and the ϵ-ANE by $\mathcal{N}_{f,\epsilon}$. Of course $\mathcal{N}_f = \mathcal{N}_{f,0} = \tilde{\mathcal{N}}_{f,0}$. These will allow us to state a result that will imply Theorem 1.2.

▶ **Theorem 4.1.**

1. *Fix a two-player game $f = (f_1, f_2)$ with strategy sets of size n_1, n_2 and all payoffs in $[-1, 1]$. Let $D = \exp(O(\ln(n_1)\ln(n_2)/\epsilon^2))$. There exists a polytope $P_f \subset \mathbb{R}^D$ such that its projection onto $\mathbb{R}^{n_1 n_2}$, called Q_f, satisfies*

   $$\mathcal{N}_f \subseteq Q_f \subseteq \mathrm{conv}(\tilde{\mathcal{N}}_{f,\epsilon}).$$

   P_f is defined by poly(D) explicit constraints and thus we can test membership in it in time poly(D).
2. *Now fix an m-player game with strategy sets of size $n_1, \dots, n_m$ and payoffs in $[-1, 1]$. Choose positive integers $k_1, \dots, k_m$. Then the same result holds with $D = n_1^{k_1} n_2^{k_2} \cdots n_m^{k_m}$ and*

   $$\epsilon = \sqrt{2 \sum_{1 \leq i < j \leq m} \frac{\ln(n_i)}{k_j}} \max_{\substack{i \in [m] \\ x \in S}} |f_i(x)|. \tag{4.2}$$

   If we specialize to $n_1 = \cdots = n_m =: n$ then we can take $D = \exp(O(m^3 \ln^2(n)/\epsilon^2))$

▶ **Corollary 4.2.** *Use the same parameters as in Theorem 4.1 and let h be an efficiently computable concave function such that $|h(p) - h(q)| \leq \eta\|p - q\|_1$ for p, q any pair of probability distributions over joint strategies. Then given some threshold T and in time $D^{O(\eta^2)}$ we can distinguish between the cases*

- $\max_{p \in \mathcal{N}_f} h(p) \geq T$*; or*
- $\max_{p \in \mathcal{N}_{f,\epsilon}} h(p) \leq T - \epsilon$*. (We could equivalently replace this with* $\max_{p \in \tilde{\mathcal{N}}_{f,\epsilon}} h(p) \leq T - \epsilon$*.)*

Corollary 4.2 follows from Theorem 4.1 and the fact that optimizing over a convex set has a poly-time reduction to the problem of testing membership in that set [13, Theorem 4.3.2 and Remark 4.2.5].

4.1 Proof for two players

While the two-player proof is a special case of the multiplayer proof, and uses very similar ideas, the notation is much simpler and it is a good warmup to the general case.

First we observe that any ϵ-correlated equilibrium q can be rounded to an ϵ-approximate NE by replacing q with the pair of marginal distributions q^X, q^Y, i.e.

$$q^X := \sum_{x \in [n_1], y \in [n_2]} \langle e_x \otimes e_y, q \rangle e_x \qquad \text{and} \qquad q^Y := \sum_{x \in [n_1], y \in [n_2]} \langle e_x \otimes e_y, q \rangle e_y \,. \tag{4.3}$$

Call this "marginal rounding."

In general marginal rounding can produce pairs of strategies that are far from equilibria. One situation in which it works well is when q is already nearly of product form; i.e. when $\|q - q^X \otimes q^Y\|_1$ is small.

▶ **Lemma 4.3.** *If q is a correlated equilibrium then (q^X, q^Y) is an ϵ-approximate NE for*

$$\epsilon = \|q - q^X \otimes q^Y\|_1 \cdot \max\{\|f_1\|_\infty, \|f_2\|_\infty\}. \tag{4.4}$$

Proof. From (2.5) we have

$$\langle q - q^X \otimes q^Y, f_1\rangle \leq \|q - q^X \otimes q^Y\|_1 \, \|f_1\|_\infty \leq \epsilon.$$

Combining with (2.3a) yields (2.2a). Repeating the argument for f_2 yields (2.2b). ◀

To obtain uncorrelated q, we will consider a variant of correlated equilibria in which there are k copies of player 2 for some positive integer k. If $q \in \Delta_{n_1 n_2^k}$ then we can interpret q as a probability distribution on random variables $X, Y_1, \ldots, Y_k$. We use the abbreviations:

$$Y_{<j} := Y_1, \ldots, Y_{j-1}$$
$$Y_{>j} := Y_{j+1}, \ldots, Y_k$$
$$Y_{-j} := Y_{<j}, Y_{>j}$$

For $y_{<j} \in [n_2]^{j-1}$, let $q^{XY_j}_{Y_{<j}=y_{<j}}$ be the distribution on XY_j obtained by conditioning on $Y_i = y_i$ for $i < j$. Explicitly

$$q^{XY_j}_{Y_{<j}=y_{<j}}(x, y_j) = \frac{q^{XY_{\leq j}}(x, y_{<j}, y_j)}{\sum_{x', y'_j} q^{XY_{\leq j}}(x', y_{<j}, y'_j)}. \tag{4.5}$$

Now define the "k-extendable relaxation" of NE to be the following LP:

$$q \in \Delta_{n_1 n_2^k} \tag{4.6a}$$
$$q^{XY_j}_{Y_{<j}=y_{<j}} \text{ satisfies (2.3)} \qquad \forall j \in [k], \forall y_{<j} \in [n_2]^{j-1} \text{ such that } q^{Y_{<j}}(y_{<j}) > 0 \tag{4.6b}$$

This is a linear program since (4.6b) is equivalent to the following uglier-but-manifestly-linear conditions:

$$\sum_{\substack{x \in [n_1] \\ y_j \in [n_2] \\ y_{>j} \in [n_2]^{k-j}}} q(x, y)(f_1(x', y_j) - f_1(x, y_j)) \leq 0 \quad \forall x' \in [n_1], \forall j \in [k], \forall y_{<j} \in [n_2]^{j-1} \tag{4.7a}$$

$$\sum_{\substack{x \in [n_1] \\ y_j \in [n_2] \\ y_{>j} \in [n_2]^{k-j}}} q(x, y)(f_2(x, y'_j) - f_2(x, y_j)) \leq 0 \quad \forall y_j \in [n_2], \forall j \in [k], \forall y_{<j} \in [n_2]^{j-1} \tag{4.7b}$$

In other words, we ask for a distribution on $XY_1 \ldots Y_k$ such that each XY_j are in a correlated equilibrium even when conditioned on the actions of $Y_{<j}$. To see that this is indeed a relaxation, observe that if p_1, p_2 is a NE, then $q = p_1 \otimes p_2^{\otimes k}$ is a valid solution to (4.6). Along with Nash's theorem, this also implies that the LP is always feasible.

The rounding algorithm is as follows.

1. Enumerate over all $j \in [k]$ and $y_{<j} \in [n]^{j-1}$.
2. For each $j, y_{<j}$, let $p_1 = q^X_{Y_{<j}=y_{<j}}$ and $p_2 = q^{Y_j}_{Y_{<j}=y_{<j}}$.
3. Output the (p_1, p_2) that is an ϵ-ANE for the lowest value of ϵ.

In the final step, we are using the fact that given p_1, p_2, it is easy to find the smallest value of ϵ for which (p_1, p_2) is an ϵ-ANE.

▶ **Theorem 4.4.** *The above procedure returns a $\sqrt{\frac{2\ln(n_1)}{k}}$-approximate NE.*

Thus, setting $k = 2\ln(n_1)/\epsilon^2$ yields an algorithm that finds an ϵ-ANE and runs in time $\text{poly}(mn^k) = \exp(O(\ln(n_1)\ln(n_2)/\epsilon^2))$. To prove Theorem 4.4 we will need a few basic facts from information theory, reviewed in Appendix C.

Proof of Theorem 4.4. Let $q^{XY_1\dots Y_k}$ be a solution of (4.6). Observe that

$$\ln(n_1) \geq H(X)_q \geq I(X;Y_1\dots Y_k)_q = \sum_{j=1}^{k} I(X;Y_j|Y_1\dots Y_{j-1}). \tag{4.8}$$

Introduce the abbreviations $\alpha = (j, y_{<j})$ for $y_{<j} \in [n_2]^{j-1}$ and $q_\alpha = q^{XY_j}|_{Y_{<j}=y_{<j}}$. Then we can rewrite (4.8) as

$$\mathop{\mathbb{E}}_{j\in[k]} \mathop{\mathbb{E}}_{y_{<j}\sim q^{Y_{<j}}} I(X;Y)_{q_\alpha} \leq \frac{\ln(n_1)}{k}. \tag{4.9}$$

Thus there is a choice of $\alpha = (j, y_{<j})$ for which $I(X;Y)_{q_\alpha} \leq \frac{\ln(n_1)}{k}$. By Pinsker's inequality (specifically (C.3)), we have

$$\|q_\alpha^{XY} - q_\alpha^X \otimes q_\alpha^Y\|_1 \leq \sqrt{\frac{2\ln(n_1)}{k}}. \tag{4.10}$$

Finally (4.6b) forces q_α to be a valid correlated equilibrium so we can use Lemma 4.3 to obtain that (q_α^X, q_α^Y) is a $\sqrt{\frac{2\ln(n_1)}{k}}$-approximate NE. ◀

In the above algorithm, we could have chosen $j, y_{<j}$ randomly instead of enumerating over all possibilities. However, this would not improve the asymptotic runtime. We also could have replaced (4.6b) with the stronger constraint that $q^{XY_j}_{Y_{-j}=y_{-j}}$ is a correlated equilibrium, with no asymptotic increase in run-time, but also without any provable performance improvement.

4.2 Proof for multiplayer games

We can quantify the distance of a distribution p to a product distribution by using the *multipartite mutual information* (which was first proposed in 1954, but has since been reinvented multiple times [23, 35, 22]

$$I(X_1 : \cdots : X_m)_p := \sum_{i=1}^{m} H(X_i)_p - H(X_1\dots X_m)_p \tag{4.11a}$$

$$= D(p^{X_1\dots X_m} \| p^{X_1} \otimes \cdots \otimes p^{X_m}) \tag{4.11b}$$

$$= \sum_{i=2}^{m} I(X_{<i} : X_i)_p \tag{4.11c}$$

Due to (4.11b) and Pinsker's inequality (see (C.1) in Appendix C), we can use the multipartite mutual information to bound the distance of a distribution to product:

$$\|p^{X_1\dots X_m} - p^{X_1} \otimes \cdots \otimes p^{X_m}\|_1 \leq \sqrt{2I(X_1 : \cdots : X_m)}. \tag{4.12}$$

To define the relaxation we will need to introduce some more notation. Let $k_1, \dots, k_m$ be positive integers that we will choose later. Define $\vec{k} = (k_1, \dots, k_m)$ and $\vec{n} = (n_1, \dots, n_m)$. Introduce random variables Y_i^j with $i \in [m]$ and $j \in [k_i]$. Let $\vec{n}^{\vec{k}} := [n_1]^{k_1} \times \cdots \times [n_m]^{k_m}$

and let $q \in \Delta_{\vec{n}^{\vec{k}}}$ be a distribution on $Y := (Y_1^1, \ldots, Y_m^{n_m})$. Define $\Phi_{\vec{k}} = [k_1] \times \cdots \times [k_m]$. If $\phi = (\phi_1, \ldots, \phi_m) \in \Phi_{\vec{k}}$, then we define

$$\begin{aligned} Y^{\phi} &:= Y_1^{\phi_1} \ldots Y_m^{\phi_m} \\ Y_{<j}^{\phi_{<j}} &:= Y_1^{\phi_1} \ldots Y_{j-1}^{\phi_{j-1}} \\ Y^{<\phi} &:= Y_1^{<\phi_1} \ldots Y_m^{<\phi_m} \\ Y_{-j}^{<\phi_{-j}} &:= Y_1^{<\phi_1} \ldots Y_{j-1}^{<\phi_{j-1}} Y_{j+1}^{<\phi_{j+1}} \ldots Y_m^{<\phi_m} \\ Y_j &:= Y_j^1 \ldots Y_j^{k_j} \end{aligned}$$

It will also be convenient to define

$$\vec{n}^{<\phi} := [n_1]^{\phi_1 - 1} \times \cdots \times [n_m]^{\phi_m - 1}$$

and likewise for $\vec{n}^{\phi}, \vec{n}^{\geq \phi}$, etc. We can now define the LP relaxation:

$$q \in \Delta_{\tilde{N}} \tag{4.13a}$$

$$q^{Y_\phi}_{Y_{<\phi} = y_{<\phi}} \text{is a correlated equilibrium} \qquad \forall \phi \in \Phi_{\vec{k}}, \forall y_{<\phi} \in \vec{n}^{<\phi} \tag{4.13b}$$

Again the constraint (4.13b) is only defined when we condition on an event with positive probability.

The rounding algorithm is similar to the bipartite case:

1. Enumerate over all $\phi \in \Phi_{\vec{k}}$ and $y_{<\phi} \in [n_1]^{\phi_1 - 1} \times \cdots [n_m]^{\phi_m - 1}$.
2. For each $\phi, y_{<\phi}$, let

$$p = \bigotimes_{i=1}^{m} q^{Y_i^{\phi_i}}_{Y_{<\phi} = y_{<\phi}}.$$

3. Output the p that is an ϵ-ANE for the lowest value of ϵ.

▶ **Theorem 4.5.** *The above procedure returns an ϵ-approximate NE, where*

$$\epsilon = \sqrt{2 \sum_{1 \leq i < j \leq m} \frac{\ln(n_i)}{k_j} \max_{\substack{i \in [m] \\ x \in S}} |f_i(x)|}. \tag{4.14}$$

Proof. We begin by bounding

$$\mathop{\mathbb{E}}_{\phi \in \Phi_{\vec{k}}} I(Y_1^{\phi_1} : \cdots : Y_m^{\phi_m} | Y^{<\phi})_p \tag{4.15a}$$

$$= \mathop{\mathbb{E}}_{\phi \in \Phi_{\vec{k}}} \sum_{j=2}^{m} I(Y_{<j}^{\phi_{<j}} : Y_j^{\phi_j} | Y^{<\phi})_p \qquad \text{using (4.11c)} \tag{4.15b}$$

$$= \sum_{j=2}^{m} \mathop{\mathbb{E}}_{\phi_{-j}} \mathop{\mathbb{E}}_{\phi_j} I(Y_{<j}^{\phi_{<j}} : Y_j^{\phi_j} | Y_j^{<\phi_j} Y_{-j}^{<\phi_{-j}})_p \qquad \text{writing } \phi = (\phi_j, \phi_{-j}) \tag{4.15c}$$

$$= \sum_{j=2}^{m} \mathop{\mathbb{E}}_{\phi_{-j}} \frac{1}{k_j} I(Y_{<j}^{\phi_{<j}} : Y_j | Y_{-j}^{<\phi_{-j}})_p \qquad \text{chain rule (C.6)} \tag{4.15d}$$

$$\leq \sum_{j=2}^{m} \frac{\ln(n_1 \cdots n_{j-1})}{k_j} = \sum_{1 \leq i < j \leq m} \frac{\ln(n_i)}{k_j} \tag{4.15e}$$

Thus there exists a ϕ and a $y^{<\phi}$ for which

$$I(Y_1^{\phi_1} : \cdots : Y_m^{\phi_m})_{p_{Y^{<\phi}=y^{<\phi}}} \leq \sum_{1\leq i<j\leq m} \frac{\ln(n_i)}{k_j}. \tag{4.16}$$

Set $q^{X_1\dots X_m} := p^{Y_1^{\phi_1}\dots Y_m^{\phi_m}}_{Y^{<\phi}=y^{<\phi}}$. Then (4.12) implies that

$$\|p^{X_1\dots X_m} - p^{X_1} \otimes \cdots \otimes p^{X_m}\|_1 \leq \sqrt{2 \sum_{1\leq i<j\leq m} \frac{\ln(n_i)}{k_j}} = \epsilon. \tag{4.17}$$

Enumerating over all ϕ will find a ϕ satisfying (4.16) and thereby also (4.17). (While this suffices for our purposes, note that concavity of the square root means that (4.17) holds in expectation even if ϕ and $y^{<\phi}$ is randomly chosen.) By an easy generalization of Lemma 4.3, we conclude that the marginal distributions of p form an ϵ-approximate Nash equilibrium. ◀

▶ **Corollary 4.6.** *For an m-player game where each player has a strategy set of size n and $\max_{i\in[n]} \max_{x\in S} |f_i(x)| \leq 1$, an ϵ-ANE can be found in time $\exp(O(m^3 \ln^2(n)/\epsilon^2))$.*

Proof. Set $k_j = \kappa\sqrt{m-j}$ in Theorem 4.5 for κ to be chosen later. The error is

$$\leq \sqrt{2\sum_{j=1}^{m} \frac{(m-j)\ln(n)}{\kappa\sqrt{m-j}}} \leq \sqrt{\frac{2m^{3/2}\ln(n)}{\kappa}}.$$

For this to be $\leq \epsilon$, we set $\kappa = \frac{2m^{3/2}\ln(n)}{\epsilon^2}$. The size of the LP is

$$n^{k_1+\dots+k_m} = \exp\left(\ln(n)\kappa \sum_{j=1}^{m} \sqrt{m-j}\right) \leq \exp\left(\frac{2m^3 \ln^2(n)}{\epsilon^2}\right),$$

and the run-time of the algorithm is polynomial in this dimension. ◀

5 Comparison to the LMM algorithm for approximate NE

Lipton, Markakis and Mehta [21] gave a method to find an ϵ-ANE in quasipolynomial time; specifically $\exp(O(\log(n_1)\log(n_2)/\epsilon^2))$. Their strategy was to prove that for any NE (p_1, p_2), there exists an ϵ-ANE $(\hat{p}_1, \hat{p}_2)$ of the form

$$\hat{p}_1 = \frac{1}{|S_X|}\sum_{x\in S_X} e_x \qquad \text{and} \qquad \hat{p}_2 = \frac{1}{|S_Y|}\sum_{y\in S_Y} e_y \tag{5.1}$$

for some multisets S_X, S_Y satisfying $|S_X| = \lceil 12\log(n_2)/\epsilon^2\rceil, |S_Y| = \lceil 12\log(n_1)/\epsilon^2\rceil$. (Their paper states a bound that is slightly worse when n_1 and n_2 are far apart, but it is not hard to improve their analysis here.) Indeed, S_X, S_Y can be obtained by randomly sampling from p_1, p_2 respectively. Such S_X, S_Y can be found deterministically by checking (2.2) for all possible choices of S_X, S_Y. This requires time

$$O(n_1^{|S_X|} n_2^{|S_Y|}) \leq O(\exp(24\log(n_1)\log(n_2)/\epsilon^2)),$$

which matches the performance of our algorithm up to the constant term in the $O()$.

In the multiplayer case, let us consider for simplicity the case of m players each with n strategies. Here LMM find that an ϵ-ANE exists with all probabilities integer multiples of $1/k$

with $k = 3m^2 \ln(m^2 n)/\epsilon^2$. The resulting runtime is $O(n^{mk}) = \exp(O(m^3 \ln(n) \ln(mn)/\epsilon^2))$. Our run-time is essentially the same, but with the $\ln(mn)$ term replaced by a $\ln(n)$ term. (However, we note that when $m \gg n$ the algorithm of [4] achieves a better runtime of $\exp(O(m(\ln(n) + \ln(m))))$.)

Our algorithm can be used to achieve a slightly different and stronger notion of approximation than [21]. Specifically, it could be used to output an ϵ-correlated equilibrium. By Lemma 4.3, this can be used to obtain an ϵ-ANE, but the reverse direction is not known.

On the other hand, if a Nash equilibrium exists with small support, then LMM will find it exactly. It does not appear that our method would take advantage of the existence of small-support Nash equilibria. Our method would outperform its worst-case bounds under a somewhat different condition: if Alice's strategy had low entropy in all correlated equilibria. Fortunately, this can be checked quickly, since correlated equilibria form a polytope and entropy is a concave function that we can maximize efficiently using standard techniques. Unfortunately, this condition does not seem to be a very natural one. As mentioned in the introduction, both methods are compatible with maximizing linear objective functions (LMM works because a Chernoff bound can also be used to show the value of the objective function can be approximated by sparse solutions), but only our method works for maximizing general concave functions, such as entropy. Entropy maximization has been discussed before in the context of games [14], but we are not aware of algorithmic implications of this.

Our algorithm also has the disadvantage (compared with LMM) of requiring quasipolynomial space, whereas LMM requires only polynomial space. On the other hand, it is possible that our LP could be approximately solved using the multiplicative weights method to reduce this space requirement.

Acknowledgments. We are grateful for helpful discussions with Costis Daskalakis, Matt Hastings and Ben Recht. AWH was funded by NSF grants CCF-1111382 and CCF-1452616. AWH and AN were funded by ARO contract W911NF-12-1-0486.

References

1 S. Aaronson, R. Impagliazzo, and D. Moshkovitz. AM with multiple merlins. In *Computational Complexity (CCC), 2014 IEEE 29th Conference on*, pages 44–55, June 2014. doi:10.1109/CCC.2014.13.

2 Noga Alon, Troy Lee, Adi Shraibman, and Santosh Vempala. The approximate rank of a matrix and its algorithmic applications: Approximate rank. In *Proceedings of the 45th Annual ACM Symposium on Symposium on Theory of Computing*, STOC'13, pages 675–684, 2013. doi:10.1145/2488608.2488694.

3 Robert Aumann. Subjectivity and correlation in randomized strategies. *Journal of Mathematical Economics*, 1(1):67–96, 1974. URL: http://EconPapers.repec.org/RePEc:eee:mateco:v:1:y:1974:i:1:p:67-96.

4 Yakov Babichenko, Siddharth Barman, and Ron Peretz. Simple approximate equilibria in large games. In *Proceedings of the Fifteenth ACM Conference on Economics and Computation*, EC'14, pages 753–770, New York, NY, USA, 2014. ACM. doi:10.1145/2600057.2602873.

5 Boaz Barak. Sum of squares upper bounds, lower bounds, and open questions. http://www.boazbarak.org/sos/files/all-notes.pdf, 2014.

6 Siddharth Barman. Approximating Nash equilibria and dense bipartite subgraphs via an approximate version of Caratheodory's theorem. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing*, STOC'15, pages 361–369, New York, NY, USA, 2015. ACM. doi:10.1145/2746539.2746566.

7 Fernando G. S. L. Brandão and Aram W. Harrow. Quantum de Finetti theorems under local measurements with applications. In *Proceedings of the 45th annual ACM Symposium on theory of computing*, STOC'13, pages 861–870, 2013.

8 Fernando G. S. L. Brandão and Aram W. Harrow. Estimating operator norms using covering nets, 2015.

9 Mark Braverman, Young Kun Ko, and Omri Weinstein. Approximating the best Nash equilibrium in $n^{o(\log n)}$-time breaks the Exponential Time Hypothesis, 2014. ECCC TR14-092.

10 Xi Chen, Xiaotie Deng, and Shang-Hua Teng. Settling the complexity of computing two-player Nash equilibria. *J. ACM*, 56(3):14:1–14:57, May 2009. doi:10.1145/1516512.1516516.

11 T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Series in Telecommunication. John Wiley and Sons, New York, 1991.

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

13 M. Grötschel, L. Lovász, and A. Schrijver. *Geometric algorithms and combinatorial optimization*. Springer-Verlag, 1993.

14 Peter D. Grünwald and A. Philip Dawid. Game theory, maximum entropy, minimum discrepancy and robust Bayesian decision theory. *Ann. Statist.*, 32(4):1367–1433, 08 2004. doi:10.1214/009053604000000553.

15 Aram W Harrow, Anand Natarajan, and Xiaodi Wu. Limitations of monogamy, Tsirelson-type bounds, and other semidefinite programs in quantum information, 2015.

16 Elad Hazan and Robert Krauthgamer. How hard is it to approximate the best nash equilibrium? *SIAM Journal on Computing*, 40(1):79–91, 2011. doi:10.1137/090766991.

17 Sam Hopkins, Pravesh Kothari, Aaron Potechin, Prasad Raghavendra, and Tselil Schramm. On the integrality gap of degree-4 sum of squares for planted clique. In *SODA'16*, 2016.

18 Jean B. Lasserre. Global optimization with polynomials and the problem of moments. *SIAM Journal on Optimization*, 11(3):796–817, 2001.

19 M. Laurent. Sums of squares, moment matrices and optimization over polynomials. *Emerging applications of algebraic geometry*, 149:157–270, 2009.

20 James R. Lee, Prasad Raghavendra, and David Steurer. Lower bounds on the size of semidefinite programming relaxations, 2014.

21 Richard J. Lipton, Evangelos Markakis, and Aranyak Mehta. Playing large games using simple strategies. In *EC'03*, pages 36–41, 2003. doi:10.1145/779928.779933.

22 Francesco M. Malvestuto. Theory of random observables in relational data bases. *Information Systems*, 8(4):281–289, 1983.

23 William J. McGill. Multivariate information transmission. *Psychometrika*, 19(2):97–116, 1954.

24 John F. Nash. Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences*, 36(1):48–49, 1950. doi:10.1073/pnas.36.1.48.

25 John F. Nash. Non-cooperative games. *Annals of mathematics*, pages 286–295, 1951.

26 Y. Nesterov. Squared functional systems and optimization problems. *High performance optimization*, 13:405–440, 2000.

27 Ryan O'Donnell, John Wright, Chenggang Wu, and Yuan Zhou. Hardness of robust graph isomorphism, lasserre gaps, and asymmetry of random graphs. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'14, pages 1659–1677. SIAM, 2014. doi:10.1137/1.9781611973402.120.

28 Christos H. Papadimitriou and Tim Roughgarden. Computing correlated equilibria in multi-player games. *J. ACM*, 55(3):14:1–14:29, August 2008. doi:10.1145/1379759.1379762.

29 Christos H. Papadimitriou and Mihalis Yannakakis. Optimization, approximation, and complexity classes. *Journal of Computer and System Sciences*, 43:425–440, 1991.

30 Pablo A. Parrilo. Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization. Technical report, MIT, 2000. Ph.D thesis.

31 Prasad Raghavendra and Ning Tan. Approximating CSPs with global cardinality constraints using SDP hierarchies. In *SODA'12*, pages 373–387, 2012.

32 Grant Schoenebeck. Linear level Lasserre lower bounds for certain k-CSPs. In *Proceedings of the 2008 49th Annual IEEE Symposium on Foundations of Computer Science*, FOCS'08, pages 593–602, Washington, DC, USA, 2008. IEEE Computer Society. doi:10.1109/FOCS.2008.74.

33 N.Z. Shor. An approach to obtaining global extremums in polynomial mathematical programming problems. *Cybernetics and Systems Analysis*, 23(5):695–700, 1987.

34 Madhur Tulsiani. Csp gaps and reductions in the lasserre hierarchy. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 303–312. ACM, 2009.

35 Satosi Watanabe. Information theoretical analysis of multivariate correlation. *IBM Journal of research and development*, 4(1):66–82, 1960.

A Polynomial Optimization and Sum-of-Squares Proofs

In this section, we lay out the basics of the sum-of-squares (SoS) optimization algorithms. They were introduced in [33, 26, 30, 18] and reviewed in [19, 5].

A.1 Polynomials

Let $\mathbb{R}[x] := \mathbb{R}[x_1, \ldots, x_n]$ be the set of real-valued polynomials over n variables, and let $\mathbb{R}[x]_d$ be the subspace of polynomials of degree $\leq d$. The set of polynomials $\mathbb{R}[x]_d$ can be viewed as $\bigoplus_{d' \leq d'} \mathrm{Sym}^{d'} \mathbb{R}^n$, where $\mathrm{Sym}^{d'} V$ denotes the symmetric subspace of $V^{\otimes d'}$.

A.2 Polynomial optimization

Given polynomials $f, g_1, \ldots, g_m \in \mathbb{R}[x]$, the basic polynomial optimization problem is to find

$$f_{\max} := \sup_{x \in \mathbb{R}^n} f(x) \text{ subject to } g_1(x) = \cdots = g_m(x) = 0. \tag{A.1}$$

Equivalently we could impose inequality constraints of the form $g_i'(x) \geq 0$ but we will not explore this option here.

A.3 Sum-of-Squares (SoS) proofs

Although (A.1) is in general NP-hard to compute exactly, the SoS hierarchy is a general method for approximating $f_{\max}$ from above. This complements simply guessing values of x or (ρ, X) which provides lower bounds on $f_{\max}$ when they satisfy the constraints. A SoS proof is a bound that makes use of the fact that $p(x)^2 \geq 0$ for any $p \in \mathbb{R}[x]$. In particular, a SoS proof that $f(x) \leq c$ for all valid f is a collection of polynomials $p_1, \ldots, p_k, q_1, \ldots, q_m \in \mathbb{R}[x]$ such that

$$c - f = \sum_{i=1}^{k} p_i^2 + \sum_{i=1}^{m} q_i g_i. \tag{A.2}$$

Observe that the RHS is ≥ 0 when evaluated on any x satisfying $g_i(x) = 0$, $\forall i$; for this reason, we refer to (A.2) as a Sum-of-Squares (SoS) proof, in particular, a proof that $c - f(x) \geq 0$ whenever $g_i(x) = 0$ for all i. This is a degree-d SoS proof if each term p_i^2 and $q_i g_i$ is in $\mathbb{R}[x]_d$. Finding an SoS proof of degree $\leq d$ can be done in time $n^{O(d)} m^{O(1)}$ using semidefinite programming [19].

If we find the minimum c for which (A.2) holds, then we obtain a hierarchy of upper bounds on $f_{\max}$, referred to as the SoS hierarchy or the Lasserre hierarchy. Denote this upper bound by f_{SoS}^d. Given mild assumptions on the constraints $g_1, \ldots, g_m$ one can prove that $\lim_{d\to\infty} f_{\text{SoS}}^d = f_{\max}$ [19]. The tradeoff between degree d and error $(f_{\text{SoS}}^d - f_{\max})$ is the key question about the SoS hierarchy. We can also express this tradeoff by defining $deg_{sos}(c - f)$ to be the minimum d for which we can find a solution to (A.2). Note that deg_{sos} has an implicit dependence on the $g_1, \ldots, g_m$.

A.4 Pseudo-expectations

We will work primarily with a dual version of SoS proofs that have an appealing probabilistic interpretation. A degree-d pseudo-expectation $\tilde{\mathbb{E}}$ is an element of $\mathbb{R}[x]_d^*$ (i.e. a linear map from $\mathbb{R}[x]_d$ to $\mathbb{R}$) satisfying

- **Normalization**. $\tilde{\mathbb{E}}[1] = 1$.
- **Positivity**. $\tilde{\mathbb{E}}[p^2] \geq 0$ for any $p \in \mathbb{R}[x]_{d/2}$.

We further say that $\tilde{\mathbb{E}}$ satisfies the constraints $g_1, \ldots, g_m$ if $\tilde{\mathbb{E}}[g_i q] = 0$ for all $i \in [n]$ and all $q \in \mathbb{R}[x]_{d-\deg(g_i)}$. Then SDP duality[3] implies that

$$f_{\text{SoS}}^d = \max\{\tilde{\mathbb{E}}[f] : \tilde{\mathbb{E}} \text{ is a degree-}d \text{ pseudo-expectation satisfying } g_1, \ldots, g_m\}. \tag{A.3}$$

The term "pseudo-expectation" comes from the fact that for any distribution μ over R^n we can define a pseudo-expectation $\tilde{\mathbb{E}}[f] := \mathbb{E}_{x\sim\mu}[f(x)]$. Thus the set of pseudo-expectations can be thought of as the low-order moments that could come from a "true" distribution μ or could come from a "fake" distribution. Indeed an alternate approach (which we will not use) proceeds from defining "pseudo-distributions" that violate the nonnegativity condition of probability distributions but in a way that cannot be detected by looking at the expectation of polynomials of degree $\leq d$ [20].

A.5 The boolean cube

Throughout this work, we will be interested in the special case of pseudo-expectations over the boolean cube $\{\pm 1\}^n$. This set is defined by the constraints $x_i^2 - 1 = 0$, $i = 1, \ldots, n$, and thus we say that $\tilde{\mathbb{E}}$ is a degree-d pseudo-expectation over $\{\pm 1\}^n$ if for any variable x_i and polynomial q of degree at most $d - 2$,

$$\tilde{\mathbb{E}}[(x_i^2 - 1)q] = 0. \tag{A.4}$$

This means we can define $\tilde{\mathbb{E}}$ entirely in terms of its action on multilinear polynomials.

[3] Certain regularity conditions (e.g. the Archimedean condition) are needed for strong duality to hold; for more details, see section 6.2 of [19]. These conditions hold in particular when the feasible set is a subset of the Boolean hypercube, which is the only setting we use in this paper.

B Framework of Deriving Lower Bounds

In this section, for the sake of completeness, we demonstrate our framework of deriving sum-of-squares (SoS) or semidefinite programming (SDP) lower bounds for optimization problems formulated in [15]. To this end, we formalize the familiar notions of optimization problem, SDP relaxations and integrality gaps. Then we show general methods for reducing optimization problems to each other as well as mapping integrality gaps for one problem/relaxation pair to another.

B.1 Optimization problems and integrality gaps

To prove a hardness result for an optimization problem, we would like to find instances where the SoS hierarchy and other SDP relaxations fail. These examples are known as "integrality gaps," where the terminology comes from the idea of approximating integer programs with convex relaxations. For our purposes, an integrality gap will be an example of an optimization (maximization) problem in which the true answer is lower than the output of the SDP relaxation. To achieve this, we need to demonstrate a feasible point of the SDP with a value that is larger than the true answer. These feasible points are called *pseudo-solutions*, and we will define them for any polynomial optimization problem as follows.

▶ **Definition B.1** (Pseudo-Solution)**.** Let A be a polynomial optimization problem. Let $\Phi_m^A \in \Delta_m^A$ be an instance of optimization A for some m. A *degree-d value-c pseudo-solution* for Φ_m^A is a degree-d pseudo-expectation $\tilde{\mathbb{E}}$ satisfying the constraints of $\mathcal{P}_n^A$ such that

$$\tilde{\mathbb{E}}[\Phi_m^A(x)] \geq c$$

A single degree-d value-c pseudo-solution for an instance Φ_m^A implies the sum-of-squares approach (up to degree d) believes the optimum value of Φ_m^A is at least c. If the true optimum value of Φ_m^A is smaller than c, then such a pseudo-solution serves as an integrality gap for the SoS approach, i.e. an example where the SoS hierarchy gives the wrong answer. To refute the power of the SoS hierarchy, we need to establish such pseudo-solutions as well as small true optimum values for any large m.

▶ **Definition B.2** (integrality gap)**.** Let A be any polynomial optimization problem. Let $d = d(n), c = c(n), s = s(n)$ be functions of n such that $0 \leq s < c \leq 1$. A degree-d value-(c, s) integrality gap for A is a collection of $\Phi_n^A \in \Delta_n^A$ for each $n \geq n_0$, s.t.

- The true optimum value $\mathrm{OPT}(\Phi_n^A) \leq s$.
- For each $n \geq n_0$, there exists a degree-d value-c pseudo-expectation $\tilde{\mathbb{E}}_n$ for Φ_n^A such that $\tilde{\mathbb{E}}_n[\Phi_n^A(x)] \geq c$.

B.2 Reduction between optimization problems

To obtain SoS lower bounds for optimization problems, it suffices to establish integrality gaps. However, it is not clear how to obtain such integrality gaps in general, which might be a challenging task by its own. Here, we formulate an approach to establish such integrality gaps through reductions. Specifically, we start with some optimization problem with known integrality gaps and reduce it to an optimization problem that we want to establish integrality gaps.

▶ **Definition B.3** (Reductions)**.** A *reduction* $R_{A \Rightarrow B}$ from optimization problem A to optimization problem B is a map from Δ^A to Δ^B; i.e. $R(\Phi_n^A) \in \Delta_n^B$. It is called

- (s^B, s^A)**-approximate** if for any n and any Φ_n^A and its corresponding $\Phi_n^B = R(\Phi_n^A)$, we have

$$\mathrm{OPT}(\Phi_n^A) = \max_{x \in \mathcal{P}_n^A} \Phi_n^A(x) \le s^A \Rightarrow \mathrm{OPT}(\Phi_n^B) = \max_{x \in \mathcal{P}_n^B} \Phi_n^B(x) \le s^B.$$

Here s^A, s^B are understood to be functions of n.

The parameters (s^B, s^A) can be considered soundness parameters of the reduction.

B.3 3XOR with integrality gap

In this section, we will introduce the source of all hardness we have for this paper, which is the 3XOR problem first discovered by Grigoriev [12] and subsequently rediscovered by Schoenebeck [32]. It is analogous to the proof that 3-SAT is NP-hard, from which other hardness results can be derived by reducing those problems to 3-SAT. In our framework, 3XOR can be formulated as follows.

▶ **Definition B.4** (3XOR)**.** 3XOR is a boolean polynomial optimization problem with the following restriction:

- **Instances**: for any n, an instance is parameterized by a formula Φ_n that consists of $m = m(n)$ 3XOR clauses, the set of which denoted by $\mathcal{C}$, on n boolean variables (i.e., each clause is $x_i x_j x_k = a_{ijk}$ for some combination of (i, j, k) and $x_i, x_j, x_k \in \{\pm 1\}$.). Thus, the objective function is

$$\Phi_n(x) = \frac{1}{m} \sum_{(i,j,k) \in \mathcal{C}} \frac{1 + a_{ijk} x_i x_j x_k}{2}, x \in \{\pm 1\}^n.$$

Thanks to the $x_i^2 = 1$ constraints, these terms are equivalent to ones of the form $(1 - (x_i x_j x_k - a_{ijk})^2)/2$.

Grigoriev's result [12] (reformulated by Barak [5]) implies the following integrality gaps. (Note that we have a slightly different formulation from [5] that is slightly stronger but guaranteed by [12].)

▶ **Proposition B.5** (Theorem 3.1 of [5], due to Grigoriev)**.** *For any $\epsilon > 0$, for every n there exists a* 3XOR *instance Φ_n with n variables and $m = O(n/\epsilon^2)$ clauses, such that $\mathrm{OPT}(\Phi_n) \le \frac{1}{2} + \epsilon$, but there exists a degree-$\Omega(n)$ value-1 pseudo-solution $\tilde{\mathbb{E}}$.*

Here "value 1" means that for every clause $x_i x_j x_k = a_{ijk}$, it holds that $\tilde{\mathbb{E}}[(x_i x_j x_k - a_{ijk}) p(x)] = 0$ for all polynomials $p(x)$ with degree at most $d - 3$.

In our framework, this implies a degree-$\Omega(n)$ value-$(1, \frac{1}{2} + \epsilon)$ integrality gap for the 3XOR problem.

C Information Theory

Proof of the claims in this section can be found in any information-theory textbook, such as [11]. For a distribution $p \in \Delta_n$ define its *entropy* to be

$$H(p) = -\sum_{x \in [n]} p(x) \ln(p(x)).$$

It can be shown that $0 \le H(p) \le \ln(n)$.

Another basic quantity is the relative entropy, which is defined for a pair of distributions $p, q \in \Delta_n$ to be

$$D(p\|q) := \sum_{x\in[n]} p(x) \ln\left(\frac{p(x)}{q(x)}\right).$$

The relative entropy has some distance-like properties. For our purposes, we will need only *Pinsker's inequality* [11, Lemma 11.6.1]:

$$D(p\|q) \geq \frac{1}{2}\|p-q\|_1^2. \tag{C.1}$$

For a distribution over several random variables e.g. $p^{XYZ} \in \Delta_{n^3}$ where X, Y, Z are supported on $[n]$, we write the entropy of the marginals using a notation that emphasizes the variables rather than the distribution:

$$H(X)_p := H(p^X), H(XY)_p := H(p^{XY}), H(XYZ)_p := H(p), \text{etc.}.$$

Using this notation we can define the mutual information between random variables X, Y to be

$$I(X;Y)_p := H(X)_p + H(Y)_p - H(XY)_p.$$

It is straightforward to show that $I(X;Y) \leq \min(H(X), H(Y))$. The mutual information is a measure of correlation, as can be seen by the following alternate characterization:

$$I(X;Y)_p = D(p^{XY}\|p^X \otimes p^Y). \tag{C.2}$$

This can be verified by a quick calculation. Combined with (C.1), we obtain

$$\|p^{XY} - p^X \otimes p^Y\|_1 \leq \sqrt{2I(X;Y)_p}. \tag{C.3}$$

Finally we will make use of the *conditional mutual information*, defined to be

$$I(X;Y|Z)_p := I(X;YZ)_p - I(X;Z)_p. \tag{C.4}$$

The term conditional mutual information refers to the alternate characterization of the quantity $I(X;Y|Z)$ as the mutual information of conditional distribution XY averaged over all values of Z; i.e.

$$I(X;Y|Z)_p = \sum_z p(Z=z) I(X;Y)|_{p_{Z=z}}, \tag{C.5}$$

where $p_{Z=z}(x,y) := p(x,y,z)/\sum_{x',y'} p(x',y',z)$.

We will also find it useful to repeatedly apply (C.4) to obtain the *chain rule of mutual information*:

$$I(X;Y_1\ldots Y_k) = \sum_{j=1}^{k} I(X;Y_j|Y_1\ldots Y_{j-1}). \tag{C.6}$$

Understanding PPA-Completeness

Xiaotie Deng[1], Jack R. Edmonds[2], Zhe Feng[3], Zhengyang Liu[1], Qi Qi[5], and Zeying Xu[6]

1 **Department of Computer Science, Shanghai Jiao Tong University, Shanghai, China**
dengxiaotie@sjtu.edu.cn
2 **Waterloo, Canada**
jack.n2m2m6@gmail.com
3 **Zhiyuan College, Shanghai Jiao Tong University, Shanghai, China**
sjtufz@sjtu.edu.cn
4 **Department of Computer Science, Shanghai Jiao Tong University, Shanghai, China**
lzy5118@sjtu.edu.cn
5 **Department of IELM, Hong Kong University of Science and Technology, Hong Kong**
kaylaqi@ust.hk
6 **Department of Mathematics, Shanghai Jiao Tong University, Shanghai, China**
zane_xu@sjtu.edu.cn

Abstract

We consider the problem of finding a fully colored base triangle on the 2-dimensional Möbius band under the standard boundary condition, proving it to be PPA-complete. The proof is based on a construction for the DPZP problem, that of finding a zero point under a discrete version of continuity condition. It further derives PPA-completeness for versions on the Möbius band of other related discrete fixed point type problems, and a special version of the Tucker problem, finding an edge such that if the value of one end vertex is x, the other is $-x$, given a special anti-symmetry boundary condition.

More generally, this applies to other non-orientable spaces, including the projective plane and the Klein bottle. However, since those models have a closed boundary, we rely on a version of the PPA that states it as to find another fixed point giving a fixed point. This model also makes it presentationally simple for an extension to a high dimensional discrete fixed point problem on a non-orientable (nearly) hyper-grid with a constant side length.

1998 ACM Subject Classification F.1.3 Complexity Measures and Classes

Keywords and phrases Fixed Point Computation, PPA-Completeness

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.23

1 Introduction

In his seminal work on understanding the time complexity of the parity argument, Papadimitriou introduced the now well known class PPAD [27] that has influenced a generation of algorithmic game theorists in their study of economic computations. In the same paper, Papadimitriou also defined a more inclusive complexity class PPA (Polynomial Parity Argument) of search problems whose solution is guaranteed to exist through a proof based on the fact that "*Any undirected graph with an odd-degree vertex must have another one*". In contrast to PPA, PPAD is based on another straightforward principle: "*Any directed graph that has an unbalanced node must have another*".

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 23; pp. 23:1–23:25

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

The class PPA is a superset of PPAD, and the intuitive reason is that *directions are helpful:* Finding another node of the appropriate kind is harder to solve when there are no directions; in fact, oracle separation is known [3]. This difference has also reflected in our understanding in the two classes, especially with regard to their complete problems. The class PPAD has now many problems that have been shown complete for PPAD such as in the incomplete list of 25 of them [22] gathered by Kintali. The class PPA-complete, however, did not fare as well.

On the one hand, there are many interesting existence theorems in Graph Theory, Combinatorics and Number Theory for which the computational problems are in PPA [27]: Smith's theorem [30] and related existentially polytime (graph) theorems [5], Chevalley's theorem [10] and Alon's Combinatorial Nullstellensatz [2], among others. Remarkably, the problem of factoring an integer has been recently proved to belong to PPA (via randomized reductions) [21], and the inclusion of this fundamental and critical problem gives the class a new significance.

On the other hand, we know few PPA-complete problems besides the generic one, unfortunately. The only exceptions are certain versions of Sperner's problem for rather esoteric non-orientable bodies. About ten years after the introduction of the class, Grigni [17] had the important idea that the right geometric context for PPA are non-orientable bodies, and showed that a version of the Sperner problem in the non-orientable three dimensional space is complete in the class. Soon after, Friedl et al. [15] strengthened it to a non-orientable and locally two-dimensional orientable space.

In general, it would be nice to have a growing strong collection of PPA-complete problems (like we have for PPAD), which with luck could eventually include factoring. The progress has been slow: another ten years passed without any progress in our understanding of the class PPA-complete for this problem many scientists are interested in.

Contributions

Our main results first end the quest for a complete fixed point characterization of the PPA-complete class. It provides a sharp division on what can be done and what cannot be done in computing different versions of the fixed point problem on the Möbius band. In particular, it does so by completing the task started by Friedl et al. [15], to reduce the next dimension demanded by the seminal result of Grigni [17], with the help of a technique developed by Chen and Deng [7], on the 2D Möbius version of a zero point problem, referred to as DPZP and conceptualised in [20, 8, 7, 11]. Together with the results of Grigni and Friedl, et al., they raise a theoretical connection of computational complexity to topology. The comparison between the 2D versions makes a strong case for this distinction.

Next, as the past works of Chen and Deng [7] as well as Deng, Qi, Saberi and Zhang [11] unify the complexity of the various discrete fixed point concepts in principle the above result implies that the same result holds for all the related discrete fixed points on the Möbius band. However, this may not always hold in general. We develop a new reduction approach to derive those results on the Möbius band. In particular, the 2D Tucker on orientable space were proven PPAD-hard, originally in the first principle by Pálvölgyi [26] and then by reduction to another discrete fixed point [11]. Both approaches are complicated where applied to the Möbius version. Our new reduction approach makes it easy to be shown in both ways of containing and contained in the PPA class. The same holds for the other discrete fixed point problems.

Third, the simplicity of our 2D version has been handy to make further applications. On the higher constant dimension non-orientable space, all the discrete fixed point problems follow from the 2D results to become PPA-complete. Those cannot be easily obtained from

the past works for the Sperner problem alone. An even bigger challenge here is whether the PPAD-completeness of the constant side length higher dimensional Sperner's problem developed by Chen, et al., [9], can be extended to the non-orientable space. Using a new (dicephalic snake) embedding lemma, together with a few demanding technical details, a 2D Sperner version is used to reduce to the higher dimension and constant side length Sperner problem on non-orientable space, and to prove the PPA-hardness of the latter. The proof involves quite some technical details but still accessible, which would be extremely difficult due to the subtlety of the boundary conditions of the non-orientable case if our Sperner on the 2D Möbius band is constructed differently. The same subtlety applies to the other discrete fixed point versions.

Fourth, the concept of the index, with modification of mod 2, is helpful both for the proofs that the above problems are in PPA, It has also be applied to develop algorithmic solutions for the oracle model of the computational problem. This approach had delivered the matching algorithmic bound for the oracle models for the fixed point problem in the orientable space [8], closing a previously almost tight gap [19]. The extension to the non-orientable space is quite natural by simply taking a mod 2 operation upon that for the orientable space. But it proves very effective. In comparison, past works have taken the path following paradigm for the fixed point computation. There are some subtleties in using index for the non-orientable space. We should not interpret the index and other values in the definitions as in the orientable space: the sense of direction no longer holds in non-orientable space at least in one dimension. Even though they are named similarly, we still need to treat them differently.

Fifth, the techniques for the 3D version may bear some similarity with our 2D version, it is exactly the articulation or, the simplicity if one prefers, in the 2 dimensional results that allows better applications to even better understanding in related problems. As we prove related results for other non-orientable spaces such as the Klein bottle or projective space, we would have to refer to less natural 3D (in 5 dimensions) Klein solid bottle or 3D projective space, with unbearable complications in the proofs. One such case is in the beautiful PSPACE proof of the other end of the line for the path following algorithm in the 2D discrete fixed point proof by Goldberg[16]. In addition, we had 20 years after the definition of PPA by Papadimitriou, 15 years after Grigni's 3D non-orientable space Sperner's PPA-completeness, and 10 years after the locally 2D Sperner's PPA-completeness by Friedl, et al. [15]. It is a time for a better understanding.

Relevance of the Möbius Band

The stories of the Möbius band have been a curiosity out of the Mind, such as a brain's toy of German mathematicians August Ferdinand Möbius (and Johann Benedict Listing), and the fascination art in *the parade of ants* by a Dutch artist M.C. Escher [13]. In recent years, it becomes a possibility in scientific discoveries. Scientists made assembled object created by nano technology [18], proposed technical tool to develop negative refractive index materials [14], made experimental observation in electromagnetic metamaterial systems [6]. In our work, it plays a role in understanding theoretical complexity of PPA-completeness. Hopefully, one day, they will become truly useful like other creatures of human imagination, if one so demands.

Related Literatures

The standard Sperner's problem, 3D-Sperner, is among the first natural problem proved to be PPAD-complete by Papadimitriou [27]. The problem 2D-Sperner is proved to be

PPAD-complete by Chen and Deng [7]. In [17], Grigni proposed the brilliant idea using non-orientable space to model the 3D-Sperner as a PPA-complete problem. The only other known PPA-complete problem is the Sperner problem on a sophisticated locally 2D structure by Friedl, Ivanyos, Santha and Verhoeven [15].

Lemke-Howson's algorithm [24] for Nash equilibrium computation has started a path following paradigm. However, a worst case exponential lower bound was known for this algorithm by Savani and von Stengel [28]. It was shown that the other PPAD-complete problems demand, under the oracle model, exponential time including the fixed point problem by Hirsch, Papadimitriou and Vavasis [19]. It was further shown to have a tight exponential time by Chen and Deng [8], which was extended to include several discrete versions of the fixed point problem by Deng, Qi, Saberi and Zhang [11].

For the PPA class, the path following method was known to take an exponential time for the Smith problem by Krawczyk [23, 4]. It has been extended to related problems, such as an exponential time bound for finding the second perfect matching on Eulerian graphs by Edmonds and Sanita [12]. An extensive discussion on related problems can be found in [5]. Subsequently, Aisenberg et. al. [1] improved our result — proving that general version for 2D-Tucker is PPA-complete using an elegant trick.

Organization of Presentation

We prove that the natural Möbius band versions of the problems, Sperner, DPZP and Tucker to be PPA-complete. A neat reduction allows the problem of finding one fixed point be extended to given-one-find-another types of PPA problems. Along with several important technical details, a dicephalic snake lemma is crucial for the padding and folding to create a higher dimensional fixed point on a non-orientable grid in order to reduce the problem to one of constant side lengths.

The paper is laid out as follows: In Section 2, we will show some necessary definitions and notations. In Section 3, we show a key result, the proof of PPA-completeness of the problem mn-DPZP and its applications. In Section 4, we extend our work to prove a high-dimensional non-orientable version of fixed point. In Section 5, we applies our main result to other non-orientable spaces, including the projective plane and the Klein bottle. We prove the PPA-completeness of the problem of finding another fixed point on the projective plane and on the Klein bottle. In Section 6, we discuss the generality of the results obtained here in related settings. Finally we discuss potential future works. Because of the space limitation, we put most of our proofs in the Appendix.

2 Preliminaries and Definitions

PPA, (in its complete form, the Polynomial Parity Argument class), is a class of search problems based on an exponential size graph consisting of nodes of maximum degree two, with a given node of degree one. The problem asks for an output of another node of degree one, which is guaranteed to exist by the parity argument. More formally, we define it by a complete problem, named AEUL as follows:

▶ **Definition 2.1** (Another End of Undirected Lines)**.** Given an input circuit T_n of polynomial size in n which takes as input u in the configuration space $C_n = \{0,1\}^n$, returns as output $T_n(u)$ in the form $\langle v, w\rangle, \langle v\rangle$, or $\langle\rangle$ where $v > w$ and $v, w \in C_n \setminus \{u\}$. $\mathbf{0}^n$ is a given configuration of one tuple, i.e., $|T_n(\mathbf{0}^n)| = 1$. The search problem is to find another configuration v, $v \neq \mathbf{0}^n$ such that $|T_n(v)| = 1$. We should write it as AEUL for short.

Möbius Band

It is obtained from a rectangle by merging its left and right sides after twisting it 180 degrees (counter)-clockwise to form a one-boundary and one-surface band. Therefore, it is non-orientable. More formally,

▶ **Definition 2.2** (Möbius Band). Let $V_{N,M} = \{\mathbf{p} = (p_1, p_2) \in \mathbb{Z}^2 : -N \le p_1 \le N, -M \le p_2 \le M\}$. A Möbius band is obtained by twisting $V_{N,M}$ 180 degrees clockwise and then joining every vertex (N, y) with $(-N, -y)$ to form a loop. We denote it by $B_{N,M}$. A function f is defined on the *Möbius band* $B_{N,M}$ iff $\forall y : -M \le y \le M$, we have $f((N, y)) = f((-N, -y))$ on $V_{N,M}$.

▶ **Definition 2.3** (Standard Triangulation). For each $i, j \in \mathbb{Z} : -N \le i < N, -M \le j < M$, we link (i, j) with $(i + 1, j + 1)$ on the grids $V_{N,M}$ and $B_{N,M}$.

We call every unit square in the standard triangulated grid $V_{N,M}$ a *base square*, every unit side length triangle of it a *base triangle*, its every edge a *base edge*.

Index

We now define the *index* [29, 31] but adopt it for the non-orientable space $B_{N,M}$.

Consider a coloring by $\{0, 1, 2\}$ of vertices in $B_{N,M}$, one vertex is assigned by one color. If a base triangle δ has all three colors, we define its index as 1. Otherwise, the index is 0. Alternatively, we define an edge index to be 1 if it is colored by both 1 and 2. The index of a base triangle is the sum of indices of its three edges, mod 2. It prepares us to define the index on Möbius band.

▶ **Definition 2.4** (Index of a Non-orientable Triangulated Möbius Grid $B_{N,M}$). Given a triangulated Möbius grid $B_{N,M}$, a coloring $\phi : B_{N,M} \to \{0, 1, 2\}$ of its vertices. The *index* of $B_{N,M}$ is defined as

$$\text{index}(B_{N,M}, \phi) := \sum_{\delta \text{ is a base triangle } \in B_{N,M}} \text{index}(\delta, \phi) \pmod 2 .$$

Immediately, one derive the following lemma about indices on the Möbius band.

▶ **Lemma 2.5.**

$$\text{index}(B_{N,M}, \phi) = \sum_{e \in \partial B_{N,M}} \text{index}(e, \phi) \pmod 2,$$

where $\partial B_{N,M}$ *is the boundary of* $B_{N,M}$*, that is,* $\partial B_{N,M} := \{(p_1, \pm M) \in \mathbb{Z}^2 : -N \le p_1 \le N\}$.

DPZP

We should introduce several concepts to prepare its definition as a numeric version of the original direction preserving zero point.

▶ **Definition 2.6** (Möbius Numeric Feasible Function). A function f: $B_{N,M} \to \{0, \pm 1, \pm 2\}$ is *feasible* if it satisfies the Möbius condition, $f((N, y)) = f((-N, -y)), \forall y \in \mathbb{Z}, -M \le y \le M$.

▶ **Definition 2.7** (Möbius Numeric Direction-preserving Function). A function f: $B_{N,M} \to \{0, \pm 1, \pm 2\}$ is *direction-preserving* if for any $\mathbf{p}, \mathbf{q} \in B_{N,M}$ where $||\mathbf{p} - \mathbf{q}||_\infty = 1$ and $f(\mathbf{p}) \neq 0$, we have $f(\mathbf{p}) + f(\mathbf{q}) \neq 0$.

▶ **Definition 2.8** (Zero Point Base Triangle)**.** Given a function $f : B_{N,M} \to \{0, \pm 1, \pm 2\}$, a base triangle δ of a triangulated Möbius Grid is called a *zero point base triangle* of f if $\{f(\mathbf{p}) : \mathbf{p} \in \delta\} = \{0, 1, 2\}$.

▶ **Definition 2.9** (Admissible Boundary Condition)**.** A function $F : B_{N,M} \to \{0, \pm 1, \pm 2\}$ is called *admissible* if it satisfied the following boundary conditions:

- $F((0, M)) = -2$; $F((0, -M)) = 2$
- $F((i, M)) = F((-i, -M)) = -1$, for every $i \in \mathbb{Z}$: $0 < i \leq N$
- $F((-i, M)) = F((i, -M)) = 1$, for every $i \in \mathbb{Z}$: $0 < i \leq N$

▶ **Definition 2.10** (Numeric Möbius DPZP)**.** Given as input, a triangulated Möbius Grid $B_{N,M}$, and a polynomial-time machine F, which generates a numeric direction-preserving feasible admissible function f on $B_{N,M}$: $f(\mathbf{p}) \in \{0, \pm 1, \pm 2\}, \forall \mathbf{p} \in B_{N,M}$, we are required to output $\mathbf{p} : f(\mathbf{p}) = 0$.

As the function $F(\cdot, \cdot)$ for mn-DPZP has five values, the index defined above does not apply. We should introduce a new definition of index for mn-DPZP.

▶ **Definition 2.11** (Index of a Base Edge and a Base Triangle in mn-DPZP)**.** Given an mn-DPZP grid $B_{N,M}$, a coloring $F : B_{N,M} \to \{0, \pm 1, \pm 2\}$, of its vertices. The *index* of an edge is 1 if the colors of its two end vertices are $\{1, 2\}$, 0 otherwise. The *index* of a base triangle is the sum of the indices of its three edges (mod 2).

▶ **Definition 2.12** (Index of mn-DPZP)**.** Given a mn-DPZP grid $B_{N,M}$, a coloring $F : B_{N,M} \to \{0, \pm 1, \pm 2\}$, of its vertices. The *index* of $B_{N,M}$ is defined as

$$\text{index}(B_{N,M}, F) := \sum_{\delta \text{ is a base triangle } \in B_{N,M}} \text{index}(\delta, F) \pmod 2 .$$

We have the following lemma on the Möbius grid.

▶ **Lemma 2.13.** $\text{index}(\delta, F) = 1$ *if and only if* $F(\delta) = \{0, 1, 2\}$. *Furthermore,* $\text{index}(B_{N,M}, F) = \sum_{e \in \partial B_{N,M}} \text{index}(e, F) \pmod 2$*, where* $\partial B_{N,M}$ *is the boundary of* $B_{N,M}$*.*

Using the index on non-orientable surfaces, it is immediately that:

▶ **Lemma 2.14.** *The Numeric Möbius DPZP with the admissible boundary always has a zero point. Finding a zero point is a PPA problem.*

We should next list the results for other related discrete fixed point concepts. We call the problem of finding a fully colored base triangle on Möbius band $B_{N,M}$ the m-Sperner problem.

▶ **Definition 2.15** (m-Sperner)**.** Consider a triangulated Möbius grid $B_{N,M}$ and a polynomial-time machine G, which generates a function g on $B_{N,M}$: $g(\mathbf{p}) = G(\mathbf{p}) \in \{0, 1, 2\}, \forall \mathbf{p} \in B_{N,M}$. Further, we require that $g(\cdot)$ satisfies the m-Sperner boundary condition, defined as follows.

- $G((0, M)) = 0$; $G((0, -M)) = 2$
- $G((i, M)) = G((-i, -M)) = 0$, for every $i \in \mathbb{Z}$: $0 < i \leq N$
- $G((-i, M)) = G((i, -M)) = 1$, for every $i \in \mathbb{Z}$: $0 < i \leq N$

The required output is a base triangle which contains all three colors.

▶ **Lemma 2.16.** *On any admissible triangulated Möbius band* $B_{N,M}$ *for an m-Sperner instance, the number of Sperner base triangles is odd. Finding one of those is in PPA.*

Proof. As m-Sperner has index 1, the oddness follows. The reduction to an AEUL is similar to the above for the mn-DPZP problem. ◀

We define the simple Möbius version of Tucker as follows.

▶ **Definition 2.17** (sm-Tucker)**.** Consider a triangulated Möbius grid $B_{N,M}$ and a polynomial-time machine G, which generates a function g on $B_{N,M}$: $g(\mathbf{p}) = G(\mathbf{p}) \in \{\pm 1, \pm 2\}, \forall \mathbf{p} \in B_{N,M}$. Further, we require that $g(\cdot)$ satisfies the special antipodal boundary condition, defined as follows:

- $g((0, M)) = -2$; $g((0, -M)) = 2$
- $g((i, M)) = g((-i, -M)) = -1$, for every $i \in \mathbb{Z}$: $0 < i \leq N$
- $g((-i, M)) = g((i, -M)) = 1$, for every $i \in \mathbb{Z}$: $0 < i \leq N$

and Möbius boundary condition which is $g((N, y)) = g((-N, -y))$. The required output is a complementary edge.

▶ **Lemma 2.18.** *On sm-Tucker, there is always a complementary edge. Finding one is in PPA.*

Proof. Changing the colors $\{-1, -2\}$ of the vertices in sm-Tucker into 0, we reduce the problem to m-Sperner. As the boundary of the m-Sperner has index 1, there is always a fully colored base triangle δ. The vertex colored 0 in δ was originally either -1 or -2 in the sm-Tucker, we obtain a complementary edge in the sm-Tucker. The claims follow. ◀

3 PPA-completeness of mn-DPZP and Its Applications

We have already proven that mn-DPZP is in PPA in the last section. We now prove the PPA-hardness of the mn-DPZP. For any input to $AEUL(T_n, C_n, 0^n)$, we construct an mn-DPZP instance in polynomial time so that each zero point in the mn-DPZP instance maps back to an end vertex for some lines in the original instance of $AEUL(T_n, C_n, 0^n)$, and vice versa.

Our proof embeds the AEUL$(T_n, C_n, 0^n)$ graph on the Möbius band. The reduction is motivated by the original proof of 2D Sperner being PPAD-complete by Chen and Deng [7].

Given a simple undirected graph $G = (V, E)$, let $|V| = N = 2^n$, we define $G^* = (V^*, E^*)$, where $V^* = V_{12N^2,24N}$. , and $E^* = \{(\mathbf{p}, \mathbf{p}') : \|\mathbf{p} - \mathbf{p}'\|_1 = 1\}$, i.e., $(\mathbf{p}, \mathbf{p}')$ is an edge in G^* if and only if their L_1 distance is 1. For every $\mathbf{p} \in V^*$, let $K_{\mathbf{p}} = \{\mathbf{q} : q_i \in \{p_i, p_i+1\}\}, i = 1, 2$ to be the vertex set containing all 4 vertices in the base square having $\mathbf{p}$ at the left bottom corner, and $E^1_{\mathbf{p}} = \{\{\mathbf{p}, \mathbf{p}+(0,1)\}, \{\mathbf{p}+(1,0), \mathbf{p}+(1,1)\}\}, E^2_{\mathbf{p}} = \{\{\mathbf{p}, \mathbf{p}+(1,0)\}, \{\mathbf{p}+(0,1), \mathbf{p}+(1,1)\}\}$ to be its two subsets of edges of $K_{\mathbf{p}}$. For $\mathbf{p}, \mathbf{q} \in V^*$, if $p_i = q_i, i = 1$ or 2, let $\mathbf{u}^1, \mathbf{u}^2, \ldots, \mathbf{u}^m \in \mathbb{Z}^2$ be all the integer internal points on segment $\mathbf{pq}$ which are labeled along $\mathbf{pq}$, where $\mathbf{u}^1 = \mathbf{p}$ and $\mathbf{u}^m = \mathbf{q}$. We say $K_{\mathbf{p}}$ and $K_{\mathbf{q}}$ are *connected* iff edges set $\cup_{k=1}^m E^i_{\mathbf{u}^k} \subseteq E^*$, we denote it by $K_{\mathbf{p}} K_{\mathbf{q}}$.

On the Möbius band, we also allow that $K_{(12N^2-1,y)}$ and $K_{(-12N^2,-y-1)}, -24N \leq y \leq 24N - 1$ can be connected, that is $E^2_{(12N^2-1,y)} \cup E^2_{(-12N^2,-y-1)} \subseteq E^*$. If we have $K_{\mathbf{u}^1} K_{\mathbf{u}^2}, K_{\mathbf{u}^2} K_{\mathbf{u}^3}, \ldots, K_{\mathbf{u}^{m-1}} K_{\mathbf{u}^m}$, but these points $\mathbf{u}^1, \mathbf{u}^2, \cdots, \mathbf{u}^m$ don't share the same x-coordinate nor y-coordinate, hence the edges introduced need to make turns in its directions to connect u^1 to u^m. We make a special note that, at a turn on K_u toward the right-upper direction, the edges $\{\{u, u + (0,1)\}, \{u + (0,1), u + (1,1)\}\}$ will be removed to make the nodes along the paths be of degree no more than two.

Intuitively, G^* is a plannar embedding of the graph G for AEUL with vertices $\{0, 1, \ldots, N-1\}$. The construction is motivated by and has some similar details to that of Chen and Deng [7]. Making it work on the Möbius band requires new ideas.

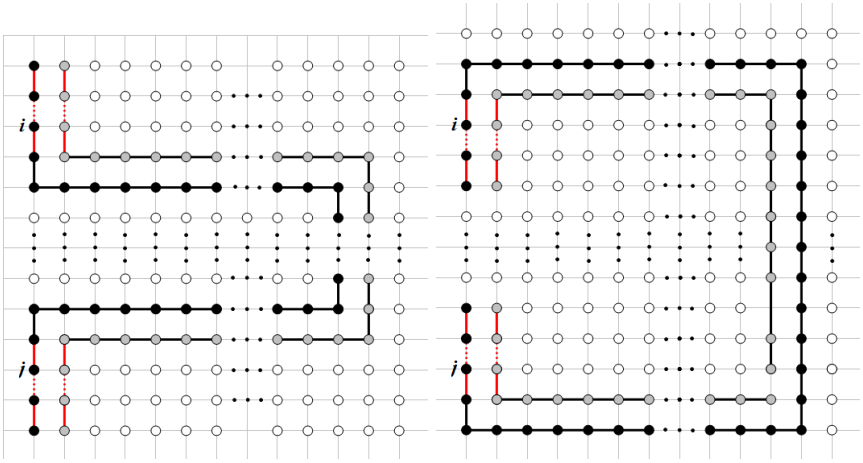


Figure 1 Connecting vertices.

For every $i : 0 \leq i < N$, vertex i of G maps to a vertex set $S_i = \cup_{k=24i}^{24i+11}\{K_{(0,k)}\}$. That is, we create a fixed-length "tube" S_i for the vertex i in G. We call it a "vertex tube". Every such tube has two ends, called *up* and *down*, dependent on their values of the second coordinates on V^*, denoted by $S_i^{up} = K_{(0,24i+11)}$ and $S_i^{down} = K_{(0,24i)}$. We make a change in the embedding of the starting node $\mathbf{0}^n$: for $i = 0$, $S_0 = \cup_{k=-24N}^{(24i+11)}\{K_{(0,k)}\}$ in G^*.

Edge ij appears in G iff there is an undirected path between one of $\{S_i^{up}, S_i^{down}\}$ and one of $\{S_j^{up}, S_j^{down}\}$. Let $ij \in E$ and $ik \in E$ be the two edges connected to j and k from i. If $j > k$ we call j the bigger neighbour and k the smaller neighbour of the vertex i.

For each vertex tube, we connect its up end to its bigger neighbour (if the degree of the vertex is 1, we also take it as the bigger one), and its down end to the smaller neighbour (if any).

If (i, j) is an edge in G, let y_i, y_j be the y-coordinates of the ends of tube i and j where need to be linked together. Let $t = 12(N \cdot \max\{i, j\} + \min\{i, j\})$. We consider two different connection cases:

1. $S_i^{up} - S_j^{down}$ or $S_i^{down} - S_j^{up}$: we add edges $K_{(0,y_i)}K_{(t,y_i)}$, $K_{(t,y_i)}K_{(t,y_j)}$, $K_{(t,y_j)}K_{(0,y_j)}$ into E^*.
2. $S_i^{up} - S_j^{up}$ or $S_i^{down} - S_j^{down}$: w.l.o.g., we assume that $i < j$, we add edges $K_{(0,y_i)}K_{(12N^2-1,y_i)}$, $K_{(12N^2-1,y_i)}K_{(-12N^2,-y_i-1)}$, $K_{(-12N^2,-y_i-1)}K_{(-t-1,-y_i-1)}$, $K_{(-t-1,-y_i-1)}K_{(-t-1,y_j)}$ and $K_{(-t-1,y_j)}K_{(0,y_j)}$ into E^*.

Case 1 is illustrated in Figure 1, which is a normal case. The crucial difference that would involve in the Möbius band structure $B_{12N^2,24N}$ is case 2, illustrated in Figure 2. For example, if degree of i is 2, i.e. $T(n, i) = \langle j, k\rangle, k > i, j$, also we assume that $i > j$ and $T(n, j) = \langle k, i\rangle$. Let $t = 10(n \cdot i + j)$, we will link S_i^{down} and S_j^{down} by adding edges $K_{(0,24i)}K_{(12N^2-1,24i)}$, $K_{(12N^2-1,24i)}K_{(-12N^2,-24i-1)}$, $K_{(-12N^2,-24i-1)}K_{(-t-1,-24i-1)}$, $K_{(-t-1,-24i-1)}K_{(-t-1,24j)}$, $K_{(-t-1,24j)}K_{(0,24j)}$.

The remaining difficulties of the reduction are how to color the vertices of G^* according to the requirements for Möbius DPZP and how to handle crossing paths. We should present techniques to handle them in the proof.

▶ **Lemma 3.1.** *mn-DPZP is PPA-hard.*

We conclude that Möbius DPZP and Möbius Tucker are PPA-complete.

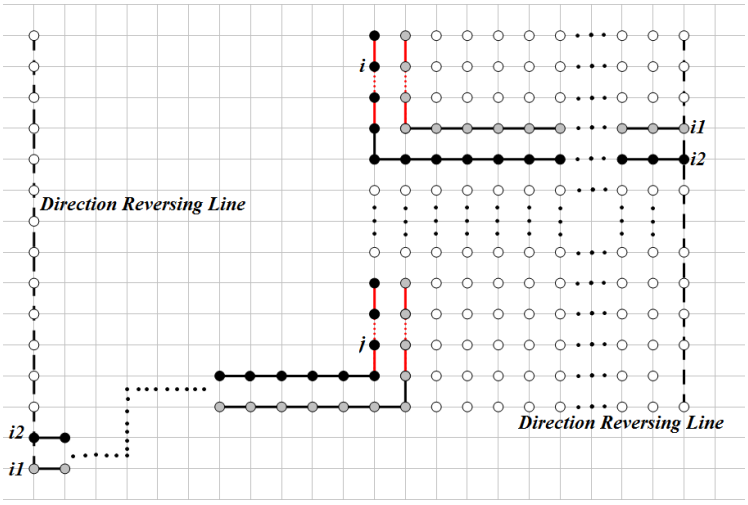


Figure 2 Connecting vertices 2.

▶ **Theorem 3.2.** *mn-DPZP is PPA-complete.*

Proof. By Lemma 2.14, mn-DPZP is in PPA. By Lemma 3.1, mn-DPZP is PPA-hard. The claim follows. ◀

▶ **Theorem 3.3.** *sm-Tucker is PPA-complete.*

Proof. sm-Tucker is in PPA by Lemma 2.18.

For PPA-hardness, we use the same construction as the proof of Lemma 3.1, except that we change vertices colored 0 to color -2. Therefore, at each vertex of color 0 in Lemma 3.1, we have an edge of color $+2$ and -2; and vice versa. The reduction follows.

Therefore, the theorem holds. ◀

Finally we show that m-Sperner is PPA-complete.

▶ **Theorem 3.4.** *m-Sperner is PPA-complete.*

Proof. First, m-Sperner is in PPA by Lemma 2.16.

To prove it is PPA-hard, we simply replace vertices colored $\{-1, -2\}$ to color 0 in the instance constructed in the PPA-hardness proof of mn-DPZP. Finding a fully colored triangle δ in the m-Sperner instance will imply a true zero point in the mn-DPZP instance because the direction preserving condition, Definition 2.7, for mn-DPZP will prevent another vertex in the same base triangle of color$\in \{-1, -2\}$.

The claim follows. ◀

4 High Dimensional Non-orientable Discrete Fixed Point

In the above, some 2D fixed point problems on the Möbius band are proven PPA-complete. The generalized problem in higher dimension space with all constant side lengths is considered in this section. The proof is motivated by a construction in [9]. To handle the non-orientable space, the key changes are on the snake lemma. We need a dicephalic snake version. Considerable changes and new ideas are required to make it through. To avoid tedious details, we should present a version of the construction and the proof. To observe the page limit, we place all the proofs and some lemmas at the appendices.

4.1 Uniform Boundary Discrete Fixed Points on Möbius Band

We introduce a version here for which the boundary of the 2D Möbius band consists vertices all of the same color. Every instance of the problem has index 0. This naturally leads to a version of the fixed point problem where one fixed point is given and another is sought after. We call such a case the uniform boundary coloring.

More precisely, the coloring function f is of *uniform boundary* on Möbius band $B_{N,M}$ if it satisfies that: (1) $f((x,\pm M)) = 0, \forall x \in \mathbb{Z}, -N \leq x \leq N$. (2) Möbius condition, i.e. $f((N,y)) = f((-N,-y)), \forall y \in \mathbb{Z}, -M \leq y \leq M$. Then the Möbius Sperner problem can be defined as follows.

▶ **Definition 4.1** (Möbius Sperner)**.** The input is a polynomial-time machine F that generates a uniform boundary 3-coloring function f on $B_{N,M}$: $F(\mathbf{p}) = f(\mathbf{p}) \in \{0,1,2\}, \forall \mathbf{p} \in B_{N,M}$, as well as a panchromatic base triangle. The required output is another panchromatic base triangle on $B_{N,M}$.

Note that $index(B_{N,M}, f)$ is zero for a color function of uniform boundary on the Möbius band. According to Lemma 2.5, we have the following lemma:

▶ **Lemma 4.2.** *For any uniform boundary 3-coloring of the triangulated Möbius band $B_{N,M}$, the number of panchromatic base triangles is even. Given one panchromatic base triangle, finding another is a PPA-complete problem.*

4.2 High Dimensional Möbius Sperner

We extend the 2-dimensional uniform boundary Möbius Sperner proven PPA-complete in the above to higher dimension. First we define the well-behaved function.

▶ **Definition 4.3** (Well-behaved Function [9])**.** A polynomial-time computable integer function f is *well-behaved*, if $\exists n_0 > 0$ such that $\forall n \geq n_0\ 3 \leq f(n) \leq n/2$.

Define $K_{\mathbf{p}} = \left\{\mathbf{q} \in \mathbb{Z}^d \mid q_i = p_i \text{ or } p_i + 1, \forall 1 \leq i \leq d\right\}$.

For a positive integer d and a vector $\mathbf{r} \in \mathbb{Z}_+^d$, let

$$A_{\mathbf{r}}^d = \left\{\mathbf{p} \in \mathbb{Z}^d \mid -r_i{+}1 \leq p_i \leq r_i{-}1, \forall 1 \leq i \leq d\right\}$$

be the *hyper grid* with side length $\mathbf{r}$ (note that is $2(r_i - 1)$ in the i-th dimension because of symmetry with respect to $r_i = 0$). Note that its *boundary* is, in one dimension, intentionally left open,

$$\partial A_{\mathbf{r}}^d = \left\{\mathbf{p} \in A_{\mathbf{r}}^d \mid p_i = -r_i{+}1 \text{ or } r_i{-}1, \exists 2 \leq i \leq d\right\}.$$

▶ **Definition 4.4** (The Valid Boundary Condition)**.** A coloring function $C : A_{\mathbf{r}}^d \to \{0,1,\ldots,d\}$ is *valid* on $A_{\mathbf{r}}^d$ if it satisfies the following Boundary Conditions:

1. (Uniform color boundary) For any $\mathbf{p} \in \partial A_r^d$, $C(\mathbf{p}) = 0$
2. (Reversing face consistency) $C((r_1{-}1, x_2, x_3, \ldots, x_d)) = C((-r_1{+}1, -x_2, x_3, \ldots, x_d))$ for all x_i, where $i = 2,3,\ldots,d$. Note that it is equivalent to merging $(r_1 - 1, x_2, x_3, \cdots, x_d)$ and $(-r_1 + 1, -x_2, x_3, \cdots, x_d)$ into one vertex.

The point set $\{(\pm(r_1{-}1), x_2, \ldots, x_d) : -r_i < x_i < r_i, i = 2,3,\ldots,d\}$ are called *reversing face*. Even though they are not on the boundary, we include (2) here to make sure the consistency of function values on the non-orientable space. Fixing other variables, $x_3, x_4, \cdots, x_d$, we have a *reversing plane* for the variables x_1 and x_2.

For any well-behaved function f, we define a corresponding Möbius-Sperner fixed point problem as follows.

▶ **Definition 4.5** (Möbius Sperner^f)**.** For a well-behaved function f and a parameter n, let $m = f(n)$ and $d = \lceil n/f(n) \rceil$. An input instance of Möbius Sperner^f is a pair $(C, 0^n)$ where C is a valid coloring function with parameter d and $\mathbf{r}$ where $r_i = 2^m, \forall i : 1 \leq i \leq d$. Given a point $\mathbf{p} \in A^d_{\mathbf{r}}$ where $K_{\mathbf{p}}$ is of degree one, i.e., contains one panchromatic simplex in its triangulation, the output of this problem is another point $\mathbf{q} \neq \mathbf{p}$, such that $K_{\mathbf{q}}$ contains another panchromatic simplex.

We have the following theorem.

▶ **Theorem 4.6.** *The problem Möbius Sperner^f is PPA-complete for any well-behaved function f.*

One can show that this problem is in PPA. To prove the hardness, similar to the orientable space [9], we embed an instance of Möbius Sperner^{f_2}, known in PPA-complete, into one dimensional higher space iteratively till Möbius Sperner^f. We should show that the process can be done in a polynomial number of state transformations. In Subsection B, we show three crucial lemmas for our reduction. In Subsection C, we employ these three lemmas iteratively to build up our construction. Please see the Appendix for the detail proofs.

5 Discrete Fixed Points on Projective Space and Klein Bottle

The results we have discussed above extend to other non-orientable spaces. The general idea is to slice out a Möbius band from the more complicated non-orientable space and to color it properly, then to patch the rest of the space. Two of the most interesting ones are the projective space and Klein Bottle. While the Möbius band can be embedded into 3D Euclidean space, neither the projective space nor the Klein bottle can. In this section, we make a reduction from DPZP to both the Möbius band and the projective plane for the PPA-hardness. As usually, as both cases are two dimensional objects, it is easy to triangulate them and to develop a path following algorithm.

We have discussed two types of discrete fixed point problems in the above. 1. finding one, and 2. (given one) finding another, dependent on the boundary conditions. As both the Projective space and the Klein bottle are closed without a boundary, we need to use the second version.

Our presentation will focus on the mn-DPZP version of the problems. The same applies to other types of discrete fixed point concepts discussed above. We omit them here as the results are similar.

▶ **Theorem 5.1.** *Given a triangulated projective plane with vertices labelled $\{0, \pm 1, \pm 2\}$. It is PPA-complete to find another zero point.*

▶ **Theorem 5.2.** *Given a triangulated Klein bottle with vertices labelled $\{0, \pm 1, \pm 2\}$. It is PPA-complete to find another zero point.*

6 Remarks and Discussion

We have discussed two types of discrete fixed point problems on the Möbius band: finding one, and (given one) finding another, dependent on the boundary conditions. We show both problems are PPA-complete for several versions of discrete fixed point models, including the Sperner's problem on the two dimensional Möbius band.

Our first step focuses on the 2D version. We start with mn-DPZP, which finds a zero point of a discrete version of the continuous functions. Based on this result, we derive PPA-completeness proof of several other related fixed point problems on the Möbius band. We discuss finding another for Möbius Sperner and Index1-Brouwer on Möbius Band. We discuss finding one for sm-Tucker and mn-DPZP. They are switchable into the other types. For example, we can change all negative colored vertices to color 0 in mn-DPZP to obtain a "finding one" version for Möbius Sperner. We leave those cases out in this version and only exemplify useful structures and techniques choosing the most typical cases.

In this work, the link between non-orientable topological space and undirected path following computational paradigm, started by Grigni in [17], is further ratified by the simple structure of 2D Möbius band. It deepens our understanding of the computational complexity difference between the two classes PPAD and PPA in terms of the underlying topological structures.

The simplicity of our construction allows itself to extend beyond the 2D Möbius band to more general cases. For example, the PPA completeness of the finding another fixed point version extends naturally to the Klein Bottle, the projective space, and to other non-orientable surfaces [32]. Simplicity has played a role in raising further curiosities from the 2D Sperner work [7] in the orientable space, such as in [25, 16].

Further the results extend to higher dimensions, even for the case where each side is of a constant length. One such high dimension non-orientable space case of finding-another fixed point is presented in Section 4. The result extends to different related solution concepts as in the previous related concepts.

Note that the discrete fixed point problems in our discussion has an exponential size configuration. Otherwise, we can enumerate the space to find a solution by brute force. To compute colors and function values, a polynomial size circuit is given as an input. Alternatively, an oracle model returns those values in a unit oracle time [19]. It is known that there is an asymptotic matching bound for finding the Brouwer's fixed point in Euclidean space [8], which extends to other discrete fixed point models [11]. The same holds for the non-orientable space we discuss here. The lower bound holds simply because the problem is harder in the non-orientable space. The upper bound follows by the standard divide-and-conquer on the index adopted for the non-orientable space.

We would like to see the natural 2D Möbius Sperner will encourage more constructive works to develop a better knowledge of the PPA-complete class. In particular, as had suggested by Grigni [17], we would like to see the computational complexity of the Smith's Theorem, known in the class of PPA, be eventually resolved.

Acknowledgements. This work was partially supported by Tianyuan Special Funds of the National Natural Science Foundation of China (No. 11426026), the National Science Foundation of China (Grant No. 61173011) and a Project 985 grant of Shanghai Jiao Tong University. Qi's work was supported by the Research Grant Council of Hong Kong (ECS Project No. 26200314 and GRF Project No. 16213115).

We would like to acknowledge our indebtedness to Xi Chen and Christos H. Papadimitriou for helps, inspirations, suggestions and discussions on PPA problems. The great appreciation is also due to the anonymous reviewers. Their constructive criticism and suggestions have been incorporated in the current revision.

References

1 James Aisenberg, Maria Luisa Bonet, and Sam Buss. 2-D Tucker is PPA complete. *ECCC TR15-163*, 2015.

2 Noga Alon. Combinatorial nullstellensatz. *Combinatorics, Probability and Computing*, 8, 1999.

3 Paul Beame, Stephen Cook, Jeff Edmonds, Russell Impagliazzo, and Toniann Pitassi. The relative complexity of NP search problems. *Journal of Computer and System Sciences*, 57(1):3–19, 1998. doi:10.1006/jcss.1998.1575.

4 Kathie Cameron. Thomason's algorithm for finding a second hamiltonian circuit through a given edge in a cubic graph is exponential on krawczyk's graphs. *Discrete Mathematics*, 235(1–3):69–77, 2001. Chech and Slovak 3. doi:10.1016/S0012-365X(00)00260-0.

5 Kathie Cameron and Jack Edmonds. Some graphic uses of an even number of odd nodes. *Annales de l'institut Fourier*, 49(3):815–827, 1999. URL: http://eudml.org/doc/75365.

6 Chih-Wei Chang, Ming Liu, Sunghyun Nam, Shuang Zhang, Yongmin Liu, Guy Bartal, and Xiang Zhang. Optical möbius symmetry in metamaterials. *Phys. Rev. Lett.*, 105:235501, Dec 2010. doi:10.1103/PhysRevLett.105.235501.

7 Xi Chen and Xiaotie Deng. On the complexity of 2D discrete fixed point problem. In *In Proceedings of the 33rd International Colloquium on Automata, Languages and Programming. Lecture Notes in Computer Science*, pages 489–500. Springer-Verlag, 2006.

8 Xi Chen and Xiaotie Deng. Matching algorithmic bounds for finding a brouwer fixed point. *JACM*, 55(3), 2008.

9 Xi Chen, Xiaotie Deng, and Shang-Hua Teng. Settling the complexity of computing two-player nash equilibria. *J. ACM*, 56(3):14:1–14:57, May 2009. doi:10.1145/1516512.1516516.

10 C. Chevalley. Démonstration d'une hypothèse de m. artin. *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, 11(1):73–75, 1935. doi:10.1007/BF02940714.

11 Xiaotie Deng, Qi Qi, Amin Saberi, and Jie Zhang. Discrete fixed points: Models, complexities, and applications. *Mathematics of Operations Research*, 36(4):636–652, 2011. doi:10.1287/moor.1110.0511.

12 Jack Edmonds and Laura Sanità. On finding another room-partitioning of the vertices. *Electronic Notes in Discrete Mathematics*, 36:1257–1264, 2010. doi:10.1016/j.endm.2010.05.159.

13 M. C. Escher. Möbius strip ii (red ants). http://www.mcescher.com/Gallery/recogn-bmp/LW441.jpg, 1963. [Online; accessed 3-March-2016].

14 Y. N. Fang, Yao Shen, Qing Ai, and C. P. Sun. Negative Refraction Induced by Möbius Topology. *ArXiv e-prints*, January 2015. URL: http://arxiv.org/abs/1501.05729.

15 Katalin Friedl, Gábor Ivanyos, Miklos Santha, and Yves F. Verhoeven. Locally 2-dimensional sperner problems complete for the polynomial parity argument classes. submitted. In *In Proceedings of the 6th Italian Conference on Algorithms and Complexity. Lecture Notes in Computer Science*, pages 380–391. Springer-Verlag, 2006.

16 Paul Goldberg. The complexity of the path-following solutions of two-dimensional spern-er/brouwer functions. *arXiv:1506.04882 [cs.CC]*, 2015.

17 Michelangelo Grigni. A sperner lemma complete for PPA. *Information Processing Letters*, 77(5–6):255–259, 2001. doi:10.1016/S0020-0190(00)00152-6.

18 Dongran Han, Suchetan Pal, Yan Liu, and Hao Yan. Folding and cutting dna into reconfigurable topological nanostructures. *Nat Nano*, 5(10):712–717, 10 2010. doi:10.1038/nnano.2010.193.

19 Michael D Hirsch, Christos H Papadimitriou, and Stephen A Vavasis. Exponential lower bounds for finding brouwer fix points. *Journal of Complexity*, 5(4):379–416, 1989. doi:10.1016/0885-064X(89)90017-4.

20 Takuya Iimura. A discrete fixed point theorem and its applications. *Journal of Mathematical Economics*, 39(7):725–742, 2003. doi:10.1016/S0304-4068(03)00007-7.

21 Emil Jerábek. Integer factoring and modular square roots. *CoRR*, abs/1207.5220, 2012. URL: http://arxiv.org/abs/1207.5220.

22 Shiva Kintali. A compendium of PPAD-complete problems. http://www.cs.princeton.edu/~kintali/ppad.html. [Online; accessed 3-March-2016].

23 Adam Krawczyk. The complexity of finding a second hamiltonian cycle in cubic graphs. *Journal of Computer and System Sciences*, 58(3):641–647, 1999. doi:10.1006/jcss.1998.1611.

24 O. L. Mangasarian. Equilibrium points of bimatrix games. *Journal of the Society for Industrial and Applied Mathematics*, 12(4):778–780, 1964. URL: http://www.jstor.org/stable/2946349.

25 Ruta Mehta. Constant rank bimatrix games are ppad-hard. In *Proceedings of the Fourty-Sixth Annual ACM Symposium on Theory of Computing*, STOC'14, pages 545–554, New York, NY, USA, 2014. ACM.

26 Dömötör Pálvölgyi. 2d-tucker is ppad-complete. In *5th International Workshop, WINE 2009, Rome, Italy, December 14-18, 2009. Proceedings*, pages 569–574. Springer-Verlag, 2009.

27 Christos H. Papadimitriou. On the complexity of the parity argument and other inefficient proofs of existence. *Journal of Computer and System Sciences*, 48(3):498–532, 1994. doi:10.1016/S0022-0000(05)80063-7.

28 Rahul Savani and Bernhard von Stengel. Hard-to-solve bimatrix games. *Econometrica*, 74(2):397–429, 2006. doi:10.1111/j.1468-0262.2006.00667.x.

29 Herbert E. Scarf. The approximation of fixed points of a continuous mapping. Cowles Foundation Discussion Papers 216R, Cowles Foundation for Research in Economics, Yale University, 1967. URL: http://EconPapers.repec.org/RePEc:cwl:cwldpp:216r.

30 A.G. Thomason. Hamiltonian cycles and uniquely edge colourable graphs. In B. Bollobás, editor, *Advances in Graph Theory*, volume 3 of *Annals of Discrete Mathematics*, pages 259–268. Elsevier, 1978. doi:10.1016/S0167-5060(08)70511-9.

31 Michael J Todd. *The computation of fixed points and applications*, volume 124. Springer Science & Business Media, 2013.

32 Eric W. Weisstein. Klein bottle. http://mathworld.wolfram.com/KleinBottle.html. [Online; accessed 3-March-2016].

A Omitted Figures and Proofs

Proof of Lemma 2.13. First, a base triangle is index 1 if and only if its vertices are colored $\{x, 1, 2\}$ where $x \notin \{1, 2\}$. As all vertices in a base triangle are of distance 1 in ∞-metric. Therefore, x can be neither -1 nor -2 by the direction preserving property. The only index 1 base triangle is $\{0, 1, 2\}$.

Next, as any internal base edge appears in the calculation of indices of two base triangles, the summation of the indices of them is either 0 or 2, which equals to $0 \pmod 2$. The claim follows. ◀

Proof of Lemma 2.14. Since we have only one edge with $(2, 1)$ on the boundary of the Möbius band, the index of edges along the boundary is 1. Therefore, by Lemma 2.13, there

is an odd number of zero point base triangles on the Möbius grid. Therefore, there is always a zero point inside the Möbius grid.

For the construction of the AEUL, we take the boundary edge $(2, 1)$ as the origin vertex of AEUL. Two such edges of mn-DPZP are connected in AEUL if they are in the same base triangle. Any such edge in mn-DPZP is an leaf node in AEUL if it is the single $\{1, 2\}$ edge in a base triangle.

Therefore, an end of lines of the AEUL instance is a base triangle of the mn-DPZP. Finding a zero point base triangle is an AEUL problem, and in PPA. ◀

Proof of Lemma 3.1. Using the main structure presented above, we show how to color $B_{12N^2,24N}$, so that for any zero point of this mn-DPZP, we can get a corresponding solution for the search problem AEUL.

The circuit T_n of AEUL generates an undirected graph $G = (C_n, E)$, where $C_n = \{0, 1\}^n$. An edge (u, v) appears in E iff $u \in T_n(v)$ and $v \in T_n(u)$.

So given any G, we construct an instance (f, G^*) for mn-DPZP problem where f is a coloring function for the generated $G^* = B_{12N^2,24N}$. We should also use $T_{12N^2,24N}$ to refer to G^* in case of no ambiguity, with the understanding that $(-12N^2, y)$ and $(12N^2, -y)$ are the same vertex.

In constructing the coloring function f for G^*, we make use of the input circuit of T_n, to identify edges connecting a node to another, and vice versa, and to identify the degree one node of the AEUL graph.

We define the coloring function as follows:

1. Color vertices on the boundary according to the admissible conditions, Definition 2.9.
2. Color the long vertex tube: $\forall j : -24N \le j < 12$, set $f((0, j)) = 2, f((1, j)) = 1$, which is a long vertex tube for the given degree one vertex $\mathbf{0}$.
3. Coat the long vertex tube (to protect positive colored 1 and 2 inside tube): $\forall j : -24N \le j < 12$, set $f((-1, j)) = -1$ and $f((2, j)) = -2$.
4. Color the other vertex tubes: $\forall i : 0 < i < N$, set $f((0, 24i + k)) = 2, f((1, 24i + k)) = 1, k = 0, 1, 2, \ldots, 11$. We need to make some modifications in the colors for the case $k = 0$ later.
5. Coat vertex tubes (to protect positive colored 1 and 2 inside tube): $\forall i : 0 < i < N$: $f((-1, 24i + k)) = -1, f((2, 24i + k)) = -2, k = 0, 1, 2, \ldots, 11$.
6. Make feasible: fill in the the rest of the interior vertices by color -2. Some of those vertices will be re-colorred in the following steps.
7. Direction preserving on end of lines: For a leaf vertex $i : 0 < i < N$, we have $f(0, 24i) = f(1, 24i) = 0$.
8. Build an edge path: Given an edge $(i, j) \in E$, w.l.o.g., assume that $i < j$, we construct a path between i and j in G^*. Let $(i', j) \in E$ and $(i, j') \in E$. If $j > j'$, then the upper end of tube for i is connected to that of j, else the lower end of the tube for i is connected to that of j. Therefore, there are four possibilities one end of the vertex tube is connected to another vertex tube.
   a. $i > i'$ and $j < j'$: Lower end of vertex tube for i is connected to the upper end of the vertex tube for j. See Figure 1.
   b. $i < i'$ and $j > j'$: Upper end of vertex tube for i is connected to the lower end of the vertex tube for j. See Figure 1.
   c. $i < i'$ and $j < j'$: Lower end of vertex tube for i is connected to the lower end of the vertex tube for j. See Figure 2.
   d. $i > i'$ and $j > j'$: Upper end of vertex tube for i is connected to the upper end of the vertex tube for j. Similar to item (c).

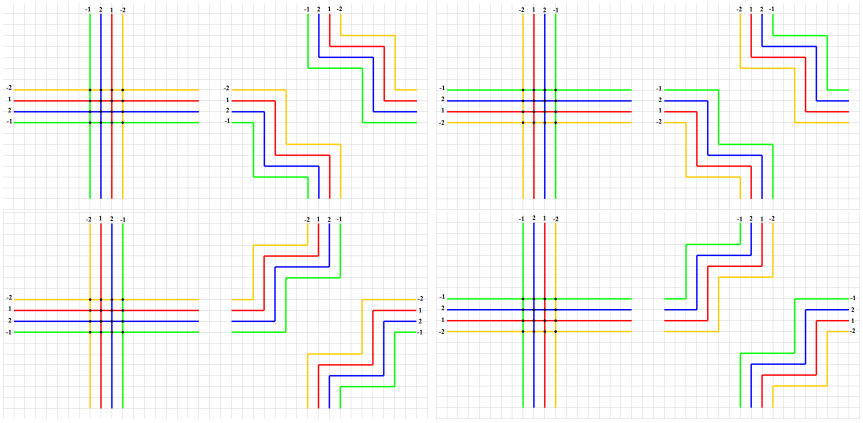


Figure 3 Connection Crossing: four cases are listed as Case 1, Case 2, Case 3, Case 4 as the normal order.

We should make appropriate adjustments so that the colorings consistently link two vertex tubes.

9. We need parallel paths of width 4, making the colors crossing it to be $\langle -1, 2, 1, -2\rangle$ (or $\langle -2, 1, 2, -1\rangle$, dependent on the direction we are moving) to maintain the direction preserving conditions. The vertex tubes for i and j connected in the four ways specified above will maintain it, that their colorings are consistent.
   Note that if the path will pass through the direction-reversing line, it must satisfy the Möbius condition, that is, the four vertices crossing the path reverse their colors from from $\langle -1, 2, 1, -2\rangle$ to $\langle -2, 1, 2, -1\rangle$ (or vice versa) after crossing the reversing direction boundary.

The colorings along the parallel paths satisfy our condition of direction preserving, as well as feasibility and admissibility conditions, except the problem where two paths cross each other. We resolve it in the same way originated from [7], shown in Figure 3. All the changes are local and can be decided using the local information with a constant bounded number of uses of the circuit T. Now we have provided the admissible coloring function that, given any point in $B_{12N^2,24N}$, provides its coloring in polynomial time using the polynomial time circuit T.

Note that vertices of color 0 in G^* only appear in the mapping from G to G^* from a vertex of degree one in G.

Therefore, finding a vertex of color 0 in G^* is equivalent to find the AEUL solution in G. Hence we have proven that mn-DPZP is PPA-hard. ◀

Proof of Lemma 4.2. Clearly, the degree of any instance is 0. Therefore, there is an even number of the fully colored base triangles. Given one fully colored Sperner base triangle, the existence of another follows by the above lemma.

The problem is in PPA because the relationship of two edges on a base triangle of colors (1,2) still holds and the uniform color boundary condition prevents the paths in the underlying AEUL going out of boundary.

On the other direction, m-Sperner can be easily reduced to Uniform-Color-Boundary Möbius Sperner by coating an extra layer of vertices outside of the boundary and coloring them all zero. More specifically, for each instance of m-Sperner, we create a Uniform-Color-Boundary Möbius Sperner by adding new vertices $\{(i, \pm(M+1)) : -N \le i \le N\}$ with all

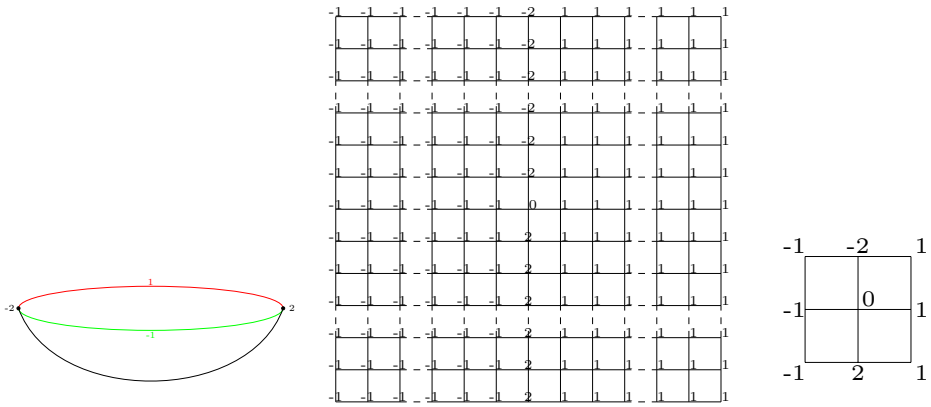


Figure 4 Disk.

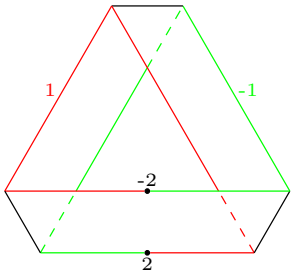


Figure 5 The Möbius Band.

color 0. After this construction, we have an instance of Uniform-Color-Boundary Möbius Sperner. There is a fully colored base triangle given $(\{(0,-M-1),(0,-M),(1,-M)\})$. Our goal is to find another which is also one for the original m-Sperner instance. ◀

The Proof of Theorem 5.1. We make a reduction from mn-DPZP to the same problem on the projective space. We will define the DPZP on the projective plane. Then, we make a reduction of mn-DPZP to the DPZP on the projective plane.

First, a 2D projective plane can be obtained from the sphere of a 3D unit ball by identifying two points share the same diameter. In other words, (x, y, z) and $(-x, -y, -z)$ in $B = \{(x, y, z) : x^2 + y^2 + z^2 = 1\}$ are merged into one point.

Next, it can be decomposed into a Möbius band and a disc as follows. $M = \{(x, y, z) : |z| \leq 1/2 : (x, y, z) \in B\}$ and $D^+ = \{(x, y, z) \in B : z \geq 1/2\}$ $D^- = \{(x, y, z) \in B : z \leq -1/2\}$. Here D^+ and D^- merge into one. We color the disc as in the central figure, at the center of the disk we place a zero point Figure 4.

Further, let $M^+ = \{(x, y, z) \in M : x \geq 0$ and $M^- = \{(x, y, z) \in M : x \leq 0$. We have M^+ is a Möbius band and so is M^-. Moreover, the interior of M^- maps into that of M^+ in a 1-1 mapping. They have a shared boundary on $x = 0$ that corresponds to the direction reversing line discussed in the above. Using the standard mn-DPZP boundary condition 2.9, we can embed an mn-DPZP instance on M^+ as in Figure 5.

Connecting the Möbius band on M^+ with the disk D^+ along their boundaries, we construct a triangulated projective plane that has a zero point on D^+ with the task of finding another zero point, which can only be on M^+. As M^+ is equivalent to an mn-DPZP instance, it follows that the task is a PPA-complete problem. ◀

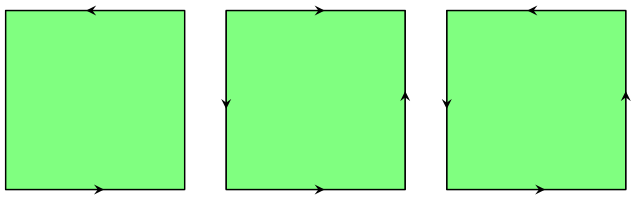


Figure 6 Grid Views: Möbius band, Klein bottle and Projective plane.

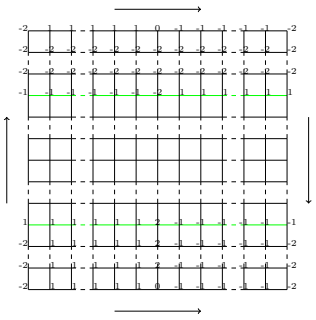


Figure 7 Embed DPZP on to Klein bottle.

The Proof of Theorem 5.2. Similarly, the PPA completeness of the finding another fixed point version extends naturally to the Klein Bottle [32]. Here again, the Klein Bottle can only be embedded in the four dimensional space. It is rather awkward to present it in the 3D world we live. Here we present by a 2D view with some amendments for ease of discussion in Figure 6. All the three non-orientable 2D spaces are represented uniformly in the 2D grid, with their boundaries merged with the opposite sides as illustrated.

This clear presentation allows a simple embedment of the DPZP grid on to the Klein bottle as presented in the following Figure 7.

Here we merge the top line and the bottom line. On it, there is a zero point in the middle. We are asked to find another zero point on the Klein bottle constructed from this grid. If we remove the top 4 lines and the bottom four lines, we obtain a mn-DPZP on the Möbius band where the only other zero points could be hidden.

Therefore, given the top zero point, finding another is to find a zero point in the mn-DPZP for the Möbius band in the middle. ◀

B Three Technical Lemmas

A triple $T = (C, d, \mathbf{r})$ is a *coloring triple* if $\mathbf{r} \in \mathbb{Z}^d$ with $r_i \geq 3$ for all $1 \leq i \leq d$ and C is a valid coloring function with parameters d and $\mathbf{r}$. Let $\text{Size}[C]$ denote the number of gates plus the number of input and output variables in a function C.

The embedding is carried out by a sequence of three polynomial-time transformations: $\mathbf{L}^1(T,\, t, u)$, $\mathbf{L}^2(T, u)$, and $\mathbf{L}^3(T, t, a, b)$. $\mathbf{L}^1(T, t, u)$ increases the t-th dimension size of the hyper grid from r_t to u (requiring $u > r_t$). $\mathbf{L}^2(T, u)$ extend the colouring into a space one dimension higher. $\mathbf{L}^3(T, t, a, b)$ folds a Möbius grid T to T' so that one more side length in a dimension is reduced to a constant size. At the same time, *from every panchromatic simplex of T', one can find a panchromatic simplex of T efficiently.* We should use $\mathbf{e}_i$ as the vector for the i-coordinate.

$\mathbf{L}^1(T,t,u)$: {Input: $T=(C,d,\mathbf{r}),t,u$} {Output: $(C',d,\mathbf{r}'),\mathbf{r}'=\mathbf{r}+(u-r_t)\mathbf{e}_t$}

1. **if** $\mathbf{p}\in\partial A^d_{\mathbf{r}'}$ **then** $C'(\mathbf{p})=0$
2. **else if** $-r_t<p_t<r_t$ **then** $C'(\mathbf{p})=C(\mathbf{p})$
3. **else** $C'(\mathbf{p})=0$

Figure 8 How $\mathbf{L}^1(T,t,u)$ extends the coloring triple $T=(C,d,\mathbf{r})$.

▶ **Lemma 2.1** ($\mathbf{L}^1(T,t,u)$: Padding a Dimension). *Given a coloring triple $T=(C,d,\mathbf{r})$ and two integers $1\le t\le d$ and $u>r_t$, $\mathbf{L}_1$ constructs a new coloring triple $T'=(C',d,\mathbf{r}')$ that satisfies the following two conditions:*

(A) *$r'_t=u$, and $r'_i=r_i$ for all other $i\in[d]$. In addition, there exists a polynomial $g_1(n)$ such that $\text{Size}[C']=\text{Size}[C]+O(g_1(\text{Size}[\mathbf{r}']))$, and T' can be computed in time polynomial in $\text{Size}[C']$. We write $T'=\mathbf{L}^1(T,t,u)$;*

(B) *From each panchromatic simplex P' of coloring triple T', we can compute a panchromatic simplex P of T in polynomial time.*

Proof. Property **A** immediately follows from Figure 8. For Property **B**, let P' be a panchromatic simplex of T', and $K_{\mathbf{p}}$ be the hypercube containing P'. We first note that $-r_t+1\le p_t<r_t-1$, because if $p_t\ge r_t-1$ or $p_t<-r_t+1$, all colors on $K_{\mathbf{p}}$ will be 0 by the color assignment. As $C'(\mathbf{q})=C(\mathbf{q})$ for all $\mathbf{q}\in A^d_{\mathbf{r}}$. Thus P' is also a panchromatic simplex of the coloring triple T. ◀

Next, we add a dimension to the grid.

▶ **Lemma 2.2** ($\mathbf{L}^2(T,u)$: Adding a Dimension). *Given a coloring triple $T=(C,d,\mathbf{r})$ and integer $u\ge 3$, $\mathbf{L}^2$ constructs a new coloring triple $T'=(C',d+1,\mathbf{r}')$ satisfying the following conditions:*

(A) *$r'_{d+1}=u$, and $r'_i=r_i$ for all $i\in[d]$. Moreover, there exists a polynomial $g_2(n)$ such that $\text{Size}[C']=\text{Size}[C]+O(g_2(\text{Size}[\mathbf{r}']))$. T' can be computed in time polynomial in $\text{Size}[C']$. We write $T'=\mathbf{L}^2(T,u)$;*

(B) *From each panchromatic simplex P' of coloring triple T', we can compute a panchromatic simplex P of T in polynomial time.*

Proof. For each point $\mathbf{p}\in A^{d+1}_{\mathbf{r}'}$, we use $\widehat{\mathbf{p}}$ to denote the point $\mathbf{z}\in A^d_{\mathbf{r}}$ with $z_i=p_i$, $\forall i\in[d]$. The color assignment of C' is given in Figure 9. Clearly, Property **A** is true.

To prove Property **B**, we let $P'\subset K_p$ be a panchromatic simplex of T'. We note that $p_{d+1}=0$. For otherwise, K_p contains color $d+1$ only if $p_{d+1}=-1$, in which case, it only contains color $d+1$ and 0, a contradiction. Therefore, the panchromatic simplex P' must be in K_p for $p_{d+1}=0$. The rest of vertices, those in $\hat{\mathbf{p}}$, must all be in K_p, which contains all the colors except $d+1$, is therefore a panchromatic simplex of T. ◀

▶ **Lemma 2.3** ($\mathbf{L}^3(T,t,a,b)$: Dicephalic Snake Embedding). *Given a coloring triple $T=(C,d,\mathbf{r})$ and integer $1\le t\le d$, if $r_t=a(2b+1)+5$ for two integers $a,b\ge 1$, then $\mathbf{L}^3$ constructs a new coloring triple $T'=(C',d+1,\mathbf{r}')$ that satisfies the following conditions:*

(A) *$r'_t=a+5$, $r'_{d+1}=4b+3$, and $r'_i=r_i$ for all other $i\in[d]$. Moreover, there exists a polynomial $g_3(n)$ such that $\text{Size}[C']=\text{Size}[C]+O(g_3(\text{Size}[\mathbf{r}']))$ and T' can be computed in time polynomial in $\text{Size}[C']$. We write $T'=\mathbf{L}^3(T,t,a,b)$.*

(B) *From each panchromatic simplex P' of coloring triple T', we can compute a panchromatic simplex P of T in polynomial time.*

$\mathbf{L}^2(T,u)$: {Input: $T=(C,d,\mathbf{r}),u$} {Output: $T'=(C',d+1,\mathbf{r}'),r'_{d+1}=u,(\forall i:1\le i\le d)r'_i=r_i$ }

1. **if** $\mathbf{p}\in\partial A^{d+1}_{r'}$ **then** $C'(\mathbf{p})=0$.
2. **else if** $p_{d+1}=1$ **then** $C'(\mathbf{p})=C(\widehat{\mathbf{p}})$ where $\widehat{\mathbf{p}}\in\mathbb{Z}^d$ satisfying $\widehat{p}_i=p_i$ for all $1\le i\le d$.
3. **else if** $p_{d+1}=0$ **then** $C'(\mathbf{p})=d+1$.
4. **else** $C'(\mathbf{p})=0$

Figure 9 How $\mathbf{L}^2(T,u)$ extends the coloring triple $T=(C,d,\mathbf{r})$.

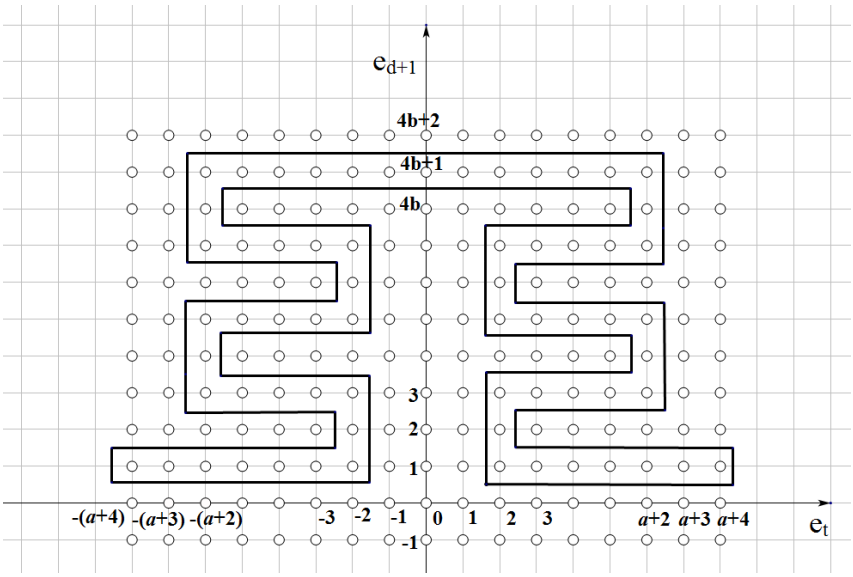


Figure 10 The two dimensional view of set $W\subset A^{d+1}_{\mathbf{r}'}$.

Proof. Consider the domains $A^d_{\mathbf{r}}\subset\mathbb{Z}^d$ and $A^{d+1}_{\mathbf{r}'}\subset\mathbb{Z}^{d+1}$ of our coloring triples. The reduction $\mathbf{L}^3(T,t,a,b)$ is carried out in three steps. First, we define a d-dimensional set $W\subset A^{d+1}_{\mathbf{r}'}$ that is large enough to contain $A^d_{\mathbf{r}}$. Second, we define a (many to one) map ψ from W to $A^d_{\mathbf{r}}$ that specifies an implicit embedding of $A^d_{\mathbf{r}}$ into W. Finally, we build a function C' for $A^{d+1}_{\mathbf{r}'}$ and show that from each panchromatic simplex of T', a panchromatic simplex of T can be found in polynomial time.

A two dimensional view of $W\subset A^{d+1}_{\mathbf{r}'}$ is illustrated in Figure 10. We use a (dicephalic) snake-pattern to realize the longer t^{th} dimension of $A^d_{\mathbf{r}}$ using the two-dimensional space defined by a new shorter t^{th} dimension and the $(d+1)^{th}$ dimension (smaller by a multiplicative factor less than one) of $A^{d+1}_{\mathbf{r}'}$, such that it is roughly $r_t=r'_t*r'_{d+1}$ (in fact, $r_t=O(r'_t*r'_{d+1})$). Formally, W consists of points $\mathbf{p}\in A^{d+1}_{\mathbf{r}'}$ satisfying $1\le p_{d+1}\le 4b+1$ and

- if $p_{d+1}=1$, then $2\le p_t\le a+4$ or $-(a+4)\le p_t\le -2$;
- if $p_{d+1}=4b+1$, then $-(a+2)\le p_t\le a+2$;
- if $p_{d+1}=4(b-i)-1$ where $0\le i\le b-1$, then $2\le p_t\le a+2$ or $-(a+2)\le p_t\le -2$;
- if $p_{d+1}=4(b-i)-3$ where $0\le i\le b-2$, then $2\le p_t\le a+2$ or $-(a+2)\le p_t\le -2$;
- if $p_{d+1}=4(b-i)-2$ where $0\le i\le b-1$, then $p_t=2$ or -2;
- if $p_{d+1}=4(b-i)$ where $0\le i\le b-1$, then $p_t=a+2$ or $-(a+2)$.

To build T', we embed the coloring triple T into W. The embedding is implicitly given by a many-to-one map ψ from W to $A^d_{\mathbf{r}}$, which will play a vital role in the coloring

$\mathbf{L}^3(T,t,a,b)$:
Input: $T=(C,d,\mathbf{r}), t, a, b,\ 1\le t\le d,\ r_t=a(2b+1)+5, a, b\ge 1$
Output: $T'=(C',d+1,\mathbf{r}'), r'_t=a+5, r'_{d+1}=4b+3, (\forall i\neq t, 1\le i\le d) r'_i=r_i$

1. **if** $\mathbf{p}\in W$ **then** $C'(\mathbf{p})=C(\psi(\mathbf{p}))$
2. **else if** $\mathbf{p}\in\partial A^{d+1}_{\mathbf{r}'}$ **then** $C'(\mathbf{p})=0$
3. **else if** $p_{d+1}=0$ **then** $C'(\mathbf{p})=d+1$.
4. **else if** $p_{d+1}=4i$ where $1\le i\le b$ and $0\le|p_t|\le a+1$ **then** $C'(\mathbf{p})=d+1$
5. **else if** $p_{d+1}=4i+1, 4i+2$ or $4i+3$ where $0\le i\le b-1$ and $|p_t|\le 1$ **then** $C'(\mathbf{p})=d+1$
6. **else** $C'(\mathbf{p})=0$

Figure 11 How $\mathbf{L}^3(T,t,a,b)$ extends the coloring triple $T=(f,d,\mathbf{r})$.

and the analysis of our reduction. For each $\mathbf{p}\in W$, we use $\mathbf{p}[m]$ to denote the point $\mathbf{q}$ in $\mathbb{Z}^d$ with $q_t=m$ and $q_i=p_i$ for all other $i\in[d]$. We denote by the function $sgn(x)=1$ if $x>0, -1$ if $x<0, 0$ if $x=0$. We define $\psi(\mathbf{p})$ according to the following cases:

- if $p_{d+1}=1$, then $\psi(\mathbf{p})=\mathbf{p}[2ab\cdot sgn(p_t)+p_t]$
- if $p_{d+1}=4b+1$, then $\psi(\mathbf{p})=\mathbf{p}[p_t]$;
- if $p_{d+1}=4(b-i)-1$ where $0\le i\le b-1$, then $\psi(\mathbf{p})=\mathbf{p}[((2i+2)a+4)\cdot sgn(p_t)-p_t]$;
- if $p_{d+1}=4(b-i)-3$ where $0\le i\le b-2$, then $\psi(\mathbf{p})=\mathbf{p}[(2i+2)a\cdot sgn(p_t)+p_t]$;
- if $p_{d+1}=4(b-i)-2$ where $0\le i\le b-1$, then $\psi(\mathbf{p})=\mathbf{p}[((2i+2)a+2)\cdot sgn(p_t)]$;
- if $p_{d+1}=4(b-i)$ where $0\le i\le b-1$, then $\psi(\mathbf{p})=\mathbf{p}[((2i+1)a+2)\cdot sgn(p_t)]$.

We let $\psi_i(\mathbf{p})$ denote the i^{th} component of $\psi(\mathbf{p})$.

▶ **Proposition 2.4** (*Valid* Boundary Condition Preserving). *The coloring function C' described in* Figure 11 *is valid on $A^{d+1}_{\mathbf{r}'}$.*

Proof. First we show that C' satisfies the uniform color boundary condition (1) for all $\mathbf{p}\in\partial A^{d+1}_{\mathbf{r}'}$. We only need to prove every vertex $\mathbf{p}\in W\cap\partial A^{d+1}_{\mathbf{r}'}$ is colored zero, by Step 2 of Figure 11.

$\forall\mathbf{p}\in W\cap\partial A^{d+1}_{\mathbf{r}'}$, by the definition of $\psi(\cdot)$, we have $p_i=\pm(r'_i-1)$ if and only if $\psi_i(\mathbf{p})=\pm(r_i-1)$ for $(i: d\ge i\ge 2)$. It follows that $C'(\mathbf{p})=C(\psi(\mathbf{p}))=0$ by the valid boundary condition for C. Therefore, C' satisfies the valid boundary condition (1).

Next we show that C' satisfies the reversing face boundary condition (2).

- If $t>2$, obviously, C' satisfies the boundary condition (2), since we have no change in x_1 nor x_2 for any set of other variables.
- If $t=1$, we consider $\mathbf{p}=(r'_1-1,x_2,x_3,\ldots,x_d,x_{d+1})$ and $\mathbf{p}'=(-r'_1+1,-x_2,x_3,\ldots,x_d,x_{d+1})$. If $x_{d+1}\neq 1, C'(\mathbf{p})=C'(\mathbf{p}')=0$. If $x_{d+1}=1$, then $\mathbf{p},\mathbf{p}'\in W$. Thus $C'(\mathbf{p})=C(\psi(\mathbf{p}))$, $C'(\mathbf{p}')=C(\psi(\mathbf{p}'))$. Since C is valid, $C(\psi(\mathbf{p}))=C(\psi(\mathbf{p}')))$ by definition of $\psi(\cdot)$. Therefore, $C'(\mathbf{p})=C'(\mathbf{p}')$.
- If $t=2$, we consider $\mathbf{p}=(r_1-1,x'_2,x_3,\ldots,x_d,x_{d+1})$ and $\mathbf{p}'=(-r_1+1,-x'_2,x_3,\ldots,x_d,x_{d+1})$. Because $\mathbf{p}$ and $\mathbf{p}'$ are central symmetric on the reversing plane, they are both in W or both not. If $\mathbf{p},\mathbf{p}'\in W$, then $C'(\mathbf{p})=C(\psi(\mathbf{p}))=C(\psi(\mathbf{p}'))=C'(\mathbf{p}')$(since C is valid). If $\mathbf{p},\mathbf{p}'$ are not in W, we have $C'(\mathbf{p})=C'(\mathbf{p}')=0$ (where p is outside W or $p_{d+1}<0$) or $C'(\mathbf{p})=C'(\mathbf{p}')=d+1$ (where p is inside W).

Therefore, C' is a valid coloring function on $A^{d+1}_{\mathbf{r}'}$. ◀

Clearly, whether $\mathbf{p} \in W$ or not can be decided in polynomial time by $\mathbf{L}^3$. Property **A** in Lemma 2.3 follows from the construction in Figure 11.

Next, we establish Property **B** of Lemma 2.3.

The intuition behind the proof is as follows. In C', vertices to the inside of W are colored in $d+1$, and vertices to the outside are colored in 0. Every (unit-size) hypercube $K_{\mathbf{p}} \subset A^{d+1}_{\mathbf{r}'}$ consists of $K_{\mathbf{p}} \cap W$, whose image $\psi(K_{\mathbf{p}} \cap W)$ is a (unit-size) hypercube in $A^d_{\mathbf{r}}$, and either vertices to the inside or the outside of W but not both. Let P' be a panchromatic simplex of T' in $A^{d+1}_{\mathbf{r}'}$. Let $K_{\mathbf{p}^*}$ be the hypercube containing P'. Since hypercubes to the outside of W do not have a vertex of color $d+1$, $K_{\mathbf{p}^*}$ must lie to the inside of W. We will show that, except the vertex of color $d+1$, every vertex $\mathbf{p} \in P'$ either belongs to $W \cap K_{\mathbf{p}^*}$, or it can be mapped to a vertex $\mathbf{q} \in W \cap K_{\mathbf{p}^*}$, such that $C'(\mathbf{q}) = C'(\mathbf{p})$. Thus from P', we can recover $d+1$ points in $W \cap K_{\mathbf{p}^*}$ with $d+1$ distinct colors $\{0, 1, \ldots, d\}$. Since $C'(\mathbf{p}) = C(\psi(\mathbf{p}))$ for all $\mathbf{p} \in W$, we can apply ψ to get a panchromatic simplex P of T.

Formally, we proceed to prove a collection of claims to cover all the possible cases of the given panchromatic simplex P' of T'. We use the following notation: For each $\mathbf{p} \in A^{d+1}_{\mathbf{r}'}$, let $\mathbf{p}[m_1, m_2]$ denote the vertex $\mathbf{q} \subset \mathbb{Z}^{d+1}$ such that $q_t = m_1$, $q_{d+1} = m_2$ and $q_i = p_i$ for all other $i \in [d]$.

▶ **Claim 2.5.** *If $p^*_t = 0$, then $p^*_{d+1} = 4b$. Furthermore, for every vertex $\mathbf{p} \in P'$ such that $C'(\mathbf{p}) \neq d+1$, $C(\psi(\mathbf{p}[p_t, 4b+1])) = C'(\mathbf{p})$.*

Proof. For the first part of the claim, we have the following contradictions if $p^*_{d+1} \neq 4b$ and $p^*_t = 0$.

1. If $p^*_{d+1} = 4b+1$, $K_{\mathbf{p}^*}$ does not contain color $d+1$.
2. $p^*_{d+1} < 0$: $C'(\mathbf{p}) \in \{0, d+1\}$, the colors of vertices in $K_{\mathbf{p}^*}$ can only be 0 or $d+1$.
3. $p^*_{d+1} < 4b$: $p^*_t = 0$ implies $p_t \in \{0, 1\}$. Therefore, each vertex $\mathbf{q} \in K_{\mathbf{p}^*}$ is colored according one of the conditions in line 3, 4, 5 or 6 of Figure 11. For each $\mathbf{q} \in K_{\mathbf{p}^*}$, $C'(\mathbf{q}) = 0$ or $d+1$ from the construction in Figure 11.

Then, $K_{\mathbf{p}^*}$ cannot be a panchromatic hypercube, contradicting the assumption of the claim. Putting these cases together, we have $p^*_{d+1} = 4b$.

We now prove the second part of the claim. If $p_{d+1} = 4b+1$, then we are done, because $C(\psi(\mathbf{p})) = C'(\mathbf{p})$ according to line 1 of Figure 11. Then $p_{d+1} = 4b$ is the only other possibility. Therefore, by the condition $C'(\mathbf{p}) \neq d+1$, according to Line 2 and 4 of Figure 11, we have $\mathbf{p} \in W \cap \partial A^{d+1}_{\mathbf{r}'}$ and $\mathbf{p}[p_t, 4b+1] = \mathbf{p}$. So we have $C(\psi(\mathbf{p}[p_t, 4b+1])) = C'(\mathbf{p}[p_t, 4b+1]) = C'(\mathbf{p})$, which completes the proof of the claim. ◀

▶ **Claim 2.6.** *If $p^*_t = a+2$ or $a+3$, then $p^*_{d+1} = 0$. In addition, for each vertex $\mathbf{p} \in P'$ such that $C'(\mathbf{p}) \neq d+1$, $C(\psi(\mathbf{p}[p_t, 1])) = C'(\mathbf{p})$.*

Proof. Obviously, $p^*_{d+1} \geq 0$. If $p^*_{d+1} > 0$, then $K_{\mathbf{p}^*}$ does not contain color $d+1$, so we have $p^*_{d+1} = 0$. The first half of the claim holds.

For the second half of the claim, first we know that if $\mathbf{p} \in W$, the claim follows. We consider the following three cases:

1. $p_{d+1} = 1$: then $\mathbf{p}$ is in W, we have $C(\psi(\mathbf{p}[p_t, 1])) = C'(\mathbf{p})$.
2. $\mathbf{p} \in A^{d+1}_{\mathbf{r}'} \setminus \partial A^{d+1}_{\mathbf{r}'}$: recall that $C'(\mathbf{q}) = d+1$ for all $\mathbf{q} \in A^{d+1}_{\mathbf{r}'} \setminus \partial A^{d+1}_{\mathbf{r}'}$ with $q_{d+1} = 0$, and we know that $p_t \in \{a+2, a+3, a+4\}$. So $\mathbf{p}$ is also in W in this case.
3. $\mathbf{p} \in \partial A^{d+1}_{\mathbf{r}'}$: we have $\mathbf{p}[p_t, 1] \in \partial A^{d+1}_{\mathbf{r}'}$, and $\psi(\mathbf{p}) = \psi(\mathbf{p}[p_t, 1])$, hence $C(\psi(\mathbf{p}[p_t, 1])) = C(\psi(\mathbf{p})) = C'(\mathbf{p})$.

Combine these three cases, the second half of the claim follows. ◀

▶ **Claim 2.7.** *If $p^*_{d+1} = 4b$, then $0 \le p^*_t \le a+1$. Moreover, for each vertex $\mathbf{p} \in P'$ such that $C'(\mathbf{p}) \neq d+1$, $C(\psi(\mathbf{p}[p_t, 4b+1])) = C'(\mathbf{p})$.*

Proof. If $p^*_t > a+1$, then $K_{\mathbf{p}^*}$ does not contain color $d+1$. So $0 \le p^*_t \le a+1$. Similar to the proof of Claim 2.5, we can prove the second part for the case when $0 \le p_t \le a+1$.

When $p_t = a+2$, both $\mathbf{p}$ and $\mathbf{p}[p_t, 4b+1]$ are in W, and we have $\psi(\mathbf{p}) = \psi(\mathbf{p}[p_t, 4b+1])$. Thus, $C(\psi(\mathbf{p}[p_t, 4b+1])) = C(\psi(\mathbf{p})) = C'(\mathbf{p})$. ◀

We can similarly prove the following claims.

▶ **Claim 2.8.** *If $p^*_{d+1} = 4i+1$ or $4i+2$ for some $0 \le i \le b-1$, then $p^*_t = 1$. Moreover, for each $\mathbf{p} \in P'$ such that $C'(\mathbf{p}) \neq d+1$, $C(\psi(\mathbf{p}[2, p_{d+1}])) = C'(\mathbf{p})$.*

▶ **Claim 2.9.** *If $p^*_{d+1} = 4i$ for some $1 \le i \le b-1$, then $1 \le p^*_t \le a+1$. In addition, for each $\mathbf{p} \in P'$ such that $C'(\mathbf{p}) \neq d+1$, if $2 \le p_t \le a+1$, then $C(\psi(\mathbf{p}[p_t, 4i+1])) = C'(\mathbf{p})$; if $p_t = 1$, then $C(\psi(\mathbf{p}[2, 4i+1])) = C'(\mathbf{p})$.*

▶ **Claim 2.10.** *If $p^*_{d+1} = 4i-1$ for some $1 \le i \le b$, then $1 \le p^*_t \le a+1$. Moreover, for each $\mathbf{p} \in P'$ such that $C'(\mathbf{p}) \neq d+1$, if $2 \le p_t \le a+1$, then $C(\psi(\mathbf{p}[p_t, 4i-1])) = C'(\mathbf{p})$; if $p_t = 1$, then $C(\psi(\mathbf{p}[2, 4i-1])) = C'(\mathbf{p})$.*

▶ **Claim 2.11.** *If $p^*_{d+1} = 0$, then $1 \le p^*_t \le a+3$. In addition, for each vertex $\mathbf{p} \in P'$ such that $C'(\mathbf{p}) \neq d+1$, if $2 \le p^*_t \le a+3$, then $\mathbf{p} \in W$ (and thus, $C(\psi(\mathbf{p})) = C'(\mathbf{p})$); if $p^*_t = 1$, then $C'\psi(\mathbf{p}[2,1])) = C'(\mathbf{p})$.*

In addition,

▶ **Claim 2.12.** $p^*_{d+1} \neq 4b+1$.

Proof. If $p^*_{d+1} = 4b+1$ then $K_{\mathbf{p}^*}$ does not contain color $d+1$. ◀

Here we do not list the cases where $p_t < 0$ where they are all the same as the above claims since W is symmetric (technically, there is one unit-sized bias between the negative and positive cases about the tth dimension). Notice that $p^*_{d+1} \ge 0$ in our construction of T'. Suppose that P' is a panchromatic simplex of T', and $K_{\mathbf{p}^*}$ be the hypercube containing P'. Then P' and $\mathbf{p}^*$ must satisfy the conditions of one of the claims above. By that claim, we can transform every vertex $\mathbf{p} \in P'$, (aside from the one that has color $d+1$) back to a vertex $\mathbf{q}$ in $A^d_{\mathbf{r}}$ to obtain a set P from P'. Since P is accommodated, it is a panchromatic simplex of t. Thus, with all the claims above, we specify an efficient algorithm to compute a panchromatic simplex P of T given a panchromatic simplex P' of T'. ◀

C Proof of The PPA-hardness in Theorem 4.6

Proof. Starting with the two dimensional case, a folding process presented next changes the size of each dimension one by one to make the size in accordance to that of the well-behaved functions. Each step uses operations $\mathbf{L}^1(T, t, u)$, $\mathbf{L}^2(T, u)$, and $\mathbf{L}^3(T, t, a, b)$ to achieve this goal and maintains the validity of boundary conditions by Lemma 2.1, 2.2 and 2.3.

The folding process from Chen et al. [9] can now be copied over by using our versions of the three basic operations, $\mathbf{L}^1$, $\mathbf{L}^2$ and $\mathbf{L}^3$, introduced above. Only some little changes are necessary in order to deal with the details for the non-orientable model. We present a simplified version here.

The Construction of $T^{3m'-14}$ from T^1

1. **for any** t from 0 to $m'-6$ **do**
2. **let** $u = (2^{(m'-t-1)(l-2)}-5)(2^{l-1}-1)+5$
3. $T^{3t+2} = \mathbf{L}^1(T^{3t+1}, 1, u)$
4. $T^{3t+3} = \mathbf{L}^3(T^{3t+2}, 1, 2^{(m'-t-1)(l-2)}, 2^{l-2}-1)$
5. $T^{3t+4} = \mathbf{L}^1(T^{3t+3}, t+3, 2^l)$

Figure 12 The Construction of $T^{3m'-14}$ from T^1.

The Construction of $T^{w'}$ from $T^{3m'-14}$

1. **let** $t = 0$
2. **while** $T^{3(m'+t)-14} = (C^{3(m'+t)-14}, m'+t-3, \mathbf{r}^{3(m'+t)-14})$ satisfies $r_1^{3(m'+t)-14} > 2^l$ **do**
3. **let** $k = \lceil (r_1^{3(m'+t)-14}-5)/(2^{l-1}-1) \rceil + 5$
4. $T^{3(m'+t)-13} = \mathbf{L}^1(T^{3(m'+t)-14}, 1, (k-5)(2^{l-1}-1)+5)$
5. $T^{3(m'+t)-12} = \mathbf{L}^3(T^{3(m'+t)-13}, 1, k, 2^{l-2}-1)$
6. $T^{3(m'+t)-11} = \mathbf{L}^1(T^{3(m'+t)-12}, m'+t-2, 2^l)$, **set** $t = t+1$
7. **let** $w' = 3(m'+t)-13$ and $T^{w'} = \mathbf{L}^1(T^{3(m'+t)-14}, 1, 2^l)$

Figure 13 The Construction of $T^{w'}$ from $T^{3m'-14}$.

Formally, let $(C, 0^{2n})$ be an input instance of Möbius Sperner^{f_2}, already proven PPA-complete. Recall that $f_2(n) = \lfloor n/2 \rfloor$. Let

$$l = f(11n) \geq 3, m' = \left\lceil \frac{n}{l-2} \right\rceil, \text{ and } m = \left\lceil \frac{11n}{l} \right\rceil.$$

For any well-behaved function f, we reduce Möbius Sperner^{f_2} to Möbius Sperner^f by iteratively constructing a sequence of coloring triple $\mathcal{T} = \{T^0, T^1, \ldots, T^w\}$ for some $w = O(m)$, where $T_0 = (C, 2, (2^n, 2^n))$ and $T_w = (C^w, m, \mathbf{r}^w)$ such that $\mathbf{r}^w \in \mathbb{Z}^m$ and $\mathbf{r}_i^w = 2^l$ for any $i, 1 \leq i \leq m$. At each phase t, we employ one of the three technical lemmas $\mathbf{L}^1, \mathbf{L}^2$ and $\mathbf{L}^3$ described in the previous subsection with appropriate parameters to construct T^{t+1} from T^t.

First, we invoke $\mathbf{L}^1\left(T^0, 1, 2^{m'(l-2)}\right)$ to get $T^1 = \left(C^1, 2, \left(2^{m'(l-2)}, 2^n\right)\right)$, where the precondition of $\mathbf{L}^1$ holds as $m'(l-2) \geq n$. Next we call the procedure in Figure 12. During every loop, the first component of $\mathbf{r}$ decreases by a factor of 2^{l-2} while the dimension of the space increases by 1 and the new dimension has a size already satisfied the requirement. So when finishing this function, we get a temporary coloring triple $T^{3m'-14} = \left(C^{3m'-14}, d^{3m'-14}, \mathbf{r}^{3m'-14}\right)$, such that

$$d^{3m'-14} = m'-3, r_1^{3m'-14} = 2^{5(l-2)}, r_2^{3m'-14} = 2^n \text{ and } r_i^{3m'-14} = 2^l, \text{ for any } i : 3 \leq i \leq m'-3.$$

Next, well invoke the procedure given in Figure 13. Note that the while-loop must terminate in at most 8 iterations because we start with $r_1^{3m'-14} = 2^{5(l-2)}$. The procedure returns a coloring triple $T^{w'} = \left(C^{w'}, d^{w'}, \mathbf{r}^{w'}\right)$ that satisfies

$$w' \leq 3m'+11, d^{w'} \leq m'+5, r_1^{w'} = 2^l, r_2^{w'} = 2^n, r_i^{w'} = 2^l, \text{ for any } i : 3 \leq i \leq d^{w'}.$$

Then we repeat the whole process above on the second coordinate and obtain a coloring triple $T^{w''} = \left(C^{w''}, d^{w''}, \mathbf{r}^{w''}\right)$ such that

$$w'' \leq 6m' + 21, d^{w''} \leq 2m' + 8 \text{ and } r_i^{w''} = 2^l, \text{ for any } i : 1 \leq i \leq d^{w''}.$$

Now follow our initial definition for m and m', we have

$$d^{w''} \leq 2m' + 8 \leq 2\left(\frac{n}{l-2} + 1\right) + 8 \leq 2\left(\frac{n}{l/3}\right) + 10 = \frac{6n}{l} + 10 \leq \frac{11n}{l} \leq m.$$

Finally, we repeat applying $\mathbf{L}^2$ for $m - d^{w''}$ times with parameter $u = 2^l$ to obtain the final coloring triple $T^w = (C^w), m, \mathbf{r}^w$ where $r_i^w = 2^l$ for any $i, 1 \leq i \leq m$. It follows our construction, $w = O(m)$.

Now we prove that the whole construction is indeed a reduction from Möbius Sperner^{f_2} to Möbius Sperner^f. Let $T^i = \left(C^i, d^i, \mathbf{r}^i\right)$, as sequence $\left\{\text{Size}\left[\mathbf{r}^i\right]\right\}_{0 \leq i \leq w}$ is non-decreasing and $w = O(m) = O(n)$, by Property **A** of Lemma 2.1, 2.2 and 2.3, there exists a polynomial $g(n)$ such that $\text{Size}\left[C^w\right] = \text{Size}\left[C\right] + O(g(n))$. By these Properties **A** again, we can construct the whole sequence $\mathcal{T}$ and in particularly, $T^w = \left(C^w, m, \mathbf{r}^2\right)$, in time polynomial in $\text{Size}\left[C\right]$.

As we know, the pair $\left(C^w, 0^{11n}\right)$ is an input instance of Möbius Sperner^f. Given a panchromatic simplex P of $\left(C^w, 0^{11n}\right)$, using the algorithm in Property **B** of Lemma 2.1, 2.2 and 2.3, we can compute a sequence of panchromatic simplex $P^w = P, P^{w-1}, \ldots, P^0$ iteratively in polynomial time, where P^t is a panchromatic simplex of T^t and can be computed from the panchromatic simplex P^{t+1} of T^{t+1}. In the end, we obtain P^0, which is a panchromatic set of $\left(C, 0^{2n}\right)$. ◀

Polynomial Bounds for Decoupling, with Applications

Ryan O'Donnell[*1] **and Yu Zhao**[†2]

1 **Computer Science Department, Carnegie Mellon University, Pittsburgh, USA**
odonnell@cs.cmu.edu@cs.cmu.edu
2 **Computer Science Department, Carnegie Mellon University, Pittsburgh, USA**
yuzhao1@cs.cmu.edu

Abstract

Let $f(x) = f(x_1, \ldots, x_n) = \sum_{|S| \le k} a_S \prod_{i \in S} x_i$ be an n-variate real multilinear polynomial of degree at most k, where $S \subseteq [n] = \{1, 2, \ldots, n\}$. For its *one-block decoupled* version,

$$\breve{f}(y, z) = \sum_{|S| \le k} a_S \sum_{i \in S} y_i \prod_{j \in S \setminus \{i\}} z_j,$$

we show tail-bound comparisons of the form

$$\mathbf{Pr}\Big[\Big|\breve{f}(\boldsymbol{y}, \boldsymbol{z})\Big| > C_k t\Big] \le D_k \, \mathbf{Pr}\Big[|f(\boldsymbol{x})| > t\Big].$$

Our constants C_k, D_k are significantly better than those known for "full decoupling". For example, when $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$ are independent Gaussians we obtain $C_k = D_k = O(k)$; when $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$ are ± 1 random variables we obtain $C_k = O(k^2)$, $D_k = k^{O(k)}$. By contrast, for full decoupling only $C_k = D_k = k^{O(k)}$ is known in these settings.

We describe consequences of these results for query complexity (related to conjectures of Aaronson and Ambainis) and for analysis of Boolean functions (including an optimal sharpening of the DFKO Inequality).

1998 ACM Subject Classification G.2 Discrete Mathematics

Keywords and phrases Decoupling, Query Complexity, Fourier Analysis, Boolean Functions

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.24

1 Introduction

Broadly speaking, *decoupling* refers to the idea of analyzing a complicated random sum involving dependent random variables by comparing it to a simpler random sum where some independence is introduced between the variables. For perhaps the simplest example, if $(a_{ij})_{i,j=1}^n \in \mathbb{R}$ and $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_n$ are independent uniform ± 1 random variables, we might ask how the moments of

$$\sum_{i,j=1}^n a_{ij} \boldsymbol{x}_i \boldsymbol{x}_j, \text{ and its "decoupled version" } \sum_{i,j=1}^n a_{ij} \boldsymbol{x}_i \boldsymbol{y}_j$$

* Supported in part by NSF grant CCF-1319743.
† Supported in part by NSF grant CCF-1319743.

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 24; pp. 24:1–24:18

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

compare. The theory of decoupling inequalities developed originally in the study of Banach spaces, stochastic processes, and U-statistics, mainly between the mid-'80s and mid-'90s; see [10] for a book-length treatment.

The powerful tool of decoupling seems to be relatively under-used in theoretical computer science. ([7] proves a variant of Hanson-Wright Inequality using decoupling inequalities with degree two; a recent work of Makarychev and Sviridenko [31] provides another exception, though they use a much different kind of decoupling than the one studied in this paper.) In this work we will observe several places where decoupling can be used in a "black-box" fashion to solve or simplify problems quite easily.

The main topic of the paper, however, is to study a partial form decoupling that we call "one-block decoupling". The advantage of one-block decoupling is that for degree-k polynomials we can achieve bounds with only *polynomial* dependence on k, as opposed to the exponential dependence on k that arises for the standard full decoupling. Although one-block decoupling does not introduce as much independence as full decoupling does, we show several applications where one-block decoupling is sufficient.

The applications we describe in this paper are the following:

- (Theorem 2.5.) Aaronson and Ambainis's conjecture concerning the generality of their [5, Theorem 4] holds. I.e., there is a sublinear-query algorithm for estimating any bounded, constant-degree Boolean function.
- (Theorem 2.8.) The Aaronson–Ambainis Conjecture [2, 4] holds if and only if it holds for one-block decoupled functions. We also show how the best known result towards the conjecture can be proven extremely easily (1) in the case of one-block decoupled functions.
- (Corollary 3.5.) An optimal form of the DFKO Fourier Tail Bound [13]: any bounded Boolean function f that is far from being a junta satisfies $\sum_{|S|>k} \widehat{f}(S)^2 \geq \exp(-O(k^2))$. Relatedly (Corollary 3.4), any degree-k real-valued Boolean function with $\Omega(1)$ variance and small influences must exceed 1 in absolute value with probability at least $\exp(-O(k^2))$; this can be further improved to $\exp(-O(k))$ if f is homogeneous.

1.1 Definitions

Throughout this section, let f denote a multilinear polynomial of degree at most k in n variables $x = (x_1, \ldots, x_n)$, with coefficients a_S from a Banach space:

$$f(x) = \sum_{\substack{S \subseteq [n] \\ |S| \leq k}} a_S x_S,$$

where we write $x_S = \prod_{i \in S} x_i$ for brevity. (The coefficients a_S will be real in all of our applications; however we allow them to be from a Banach space since the proofs are no more complicated.)

We begin by defining our notion of partial decoupling:

▶ **Definition 1.1.** The *one-block decoupled* version of f, denoted $\breve{f}$, is the multilinear polynomial over $2n$ variables $y = (y_1, \ldots, y_n)$ and $z = (z_1, \ldots, z_n)$ defined by

$$\breve{f}(y, z) = \sum_{\substack{S \subseteq [n] \\ 1 \leq |S| \leq k}} a_S \sum_{i \in S} y_i z_{S \setminus i}.$$

In other words, each monomial term like $x_1 x_3 x_7$ is replaced with $y_1 z_3 z_7 + z_1 y_3 z_7 + z_1 z_3 y_7$. In case f is homogeneous we have the relation $\breve{f}(x, x) = k f(x)$.

Let us also recall the traditional notion of decoupling:

▶ **Definition 1.2.** The *(fully) decoupled* version of f, which we denote by $\widetilde{f}$, is a multilinear polynomial over k *blocks* $x^{(1)}, \dots, x^{(k)}$ of n variables; each $x^{(i)}$ is $x^{(i)} = (x_1^{(i)}, \dots, x_n^{(i)})$. It is formed as follows: for each monomial x_S in f, we replace it with the average over all ways of assigning its variables to different blocks. More formally,

$$\widetilde{f}(x^{(1)}, \dots, x^{(k)}) = a_\emptyset + \sum_{\substack{S \subseteq [n] \\ 1 \le |S| \le k}} \frac{(k - |S|)!}{k!} \cdot a_S \sum_{\substack{\text{injective} \\ b : S \to [k]}} \prod_{i \in S} x_i^{(b(i))}.$$

The definition is again simpler if f is homogeneous. For example, if f is homogeneous of degree 3, then each monomial in f like $x_1 x_3 x_7$ is replaced in $\widetilde{f}$ with

$$\frac{1}{6}\left(w_1 y_2 z_3 + w_1 z_2 y_3 + y_1 w_2 z_3 + y_1 z_2 w_3 + z_1 w_2 y_3 + z_1 y_2 w_3\right).$$

(Here we wrote w, y, z instead of $x^{(1)}$, $x^{(2)}$, $x^{(3)}$, for simplicity.) Note that $\widetilde{f}(x, x, \dots, x) = f(x)$ always holds, even if f is not homogeneous.

We conclude by comparing the two kinds of decoupling. Assume for simplicity that f is homogeneous of degree k. The fully decoupled version $\widetilde{f}(x^{(1)}, \dots, x^{(k)})$ is in "block-multilinear form"; i.e., each monomial contains exactly one variable from each of the k "blocks". This kind of structure has often been recognized as useful in theoretical computer science; see, e.g., [24, 29, 21, 5]. By contrast, the one-block decoupling $\breve{f}(y, z)$ does not have such a simple structure; we only have that each monomial contains exactly one y-variable. Nevertheless we will see several examples in this paper where having one-block decoupled form is just as useful as having fully decoupled form. And as mentioned, we will show that it is possible to achieve one-block decoupling with only poly(k) parameter losses, whereas full decoupling in general suffers exponential losses in k.

▶ Remark 1.1. We have also chosen different "scalings" for the two kinds of decoupling. For example, in the homogeneous case, we have $\widetilde{f}(y, z, z, \dots, z) = \frac{1}{k} \cdot \breve{f}(y, z)$ and also $\mathbf{Var}[\widetilde{f}] = \frac{1}{k \cdot k!} \mathbf{Var}[\breve{f}]$ for $f : \{\pm 1\}^n \to \mathbb{R}$.

1.2 A useful inequality

Several times we will use the following basic inequality from analysis of Boolean functions, which relies on hypercontractivity; see [33, Theorems 9.24, 10.23].

▶ **Theorem 1.3.** *Let $f(x) = \sum_{|S| \le k} a_S x_S$ be a nonconstant n-variate multilinear polynomial of degree at most k, where the coefficients a_S are real. Let $\boldsymbol{x}_1, \dots, \boldsymbol{x}_n$ be independent uniform ± 1 random variables. Then*

$$\mathbf{Pr}\big[f(\boldsymbol{x}) > \mathbf{E}[f]\big] \ge \tfrac{1}{4} e^{-2k}.$$

This also holds if some of the $\boldsymbol{x}_i$'s are standard Gaussians.[1] *Finally, if the $\boldsymbol{x}_i$'s are not uniform ± 1 random variables, but they take on each value ± 1 with probability at least λ, then we may replace $\frac{1}{4} e^{-2k}$ by $\frac{1}{4}(e^2/2\lambda)^{-k}$.*

[1] Although it is not stated in [33], an identical proof works since Gaussians have the same hypercontractivity properties as uniform ± 1 random variables.

2 Decoupling theorems, and query complexity applications

2.1 Classical decoupling inequalities, and an application in query complexity

Traditional decoupling inequalities compare the probabilistic behavior of f and $\widetilde{f}$ under independent random variables (usually symmetric ones; e.g., standard Gaussians). The easier forms of the inequalities compare expectations under a convex test function; e.g., they can be used to compare p-norms. The following was essentially proved in [9]; see [10, Theorem 3.1.1,(3.4.23)–(3.4.27)]:

▶ **Theorem 2.1.** *Let $\Phi : \mathbb{R}^{\geq 0} \to \mathbb{R}^{\geq 0}$ be convex and nondecreasing. Let $\boldsymbol{x} = (\boldsymbol{x}_1, \dots, \boldsymbol{x}_n)$ consist of independent real random variables with all moments finite, and let $\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(k)}$ denote independent copies of $\boldsymbol{x}$. Then*

$$\mathbf{E}\Big[\Phi\Big(\Big\|\widetilde{f}\left(\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(k)}\right)\Big\|\Big)\Big] \leq \mathbf{E}\Big[\Phi\Big(C_k \left\|f\left(\boldsymbol{x}\right)\right\|\Big)\Big],$$

where $C_k = k^{O(k)}$ is a constant depending only on k.

▶ Remark. A reverse inequality also holds, with worse constant $C_k = k^{-O(k^2)}$.

Another line of research gave comparisons between tail bounds for f and $\widetilde{f}$. This culminated in the following theorem from [11, 18]; see also [10, Theorem 3.4.6]:

▶ **Theorem 2.2.** *In the setting of Theorem 2.1, for all $t > 0$,*

$$\mathbf{Pr}\Big[\Big\|\widetilde{f}\left(\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(k)}\right)\Big\| > C_k t\Big] \leq D_k \, \mathbf{Pr}\Big[\left\|f\left(\boldsymbol{x}\right)\right\| > t\Big],$$

where $C_k = D_k = k^{O(k)}$. The analogous reverse bound also holds.

▶ Remark 2.1. Kwapień [28] showed that when the $\boldsymbol{x}_i$'s are α-stable random variables, the constant C_k in Theorem 2.1, can be improved to $k^{k/\alpha}/k!$; this is $k^{k/2}/k!$ for standard Gaussians. Furthermore, for uniform ± 1 random variables Kwapień's proof goes through as if they were 1-stable; thus in this case one may take $C_k = k^k/k! \leq e^k$. In the Gaussian setting with homogeneous f, Kwapień obtains $C_k = k^{k/2}/k!$ and $D_k = 2^k$ for Theorem 2.2.

For function $f(\boldsymbol{x}) = \sum_{|S| \leq k} a_S \boldsymbol{x}_S$ where coefficients a_S are real, we denote its p-norm $\|f\|_p = \mathbf{E}[f(\boldsymbol{x})^p]^{1/p}$. Furthermore if f is a bounded function with input $\boldsymbol{x}$, we denote the infinity norm

$$\|f\|_\infty = \lim_{p \to \infty} \|f\|_p = \sup_{\boldsymbol{x}} |f(\boldsymbol{x})|.$$

▶ **Corollary 2.3.** *In the setting of Theorem 2.1, it holds that $\|\widetilde{f}\|_\infty \leq k^{O(k)} \|f\|_\infty$. Further, if $f : \{\pm 1\}^n \to \mathbb{R}$ then $\|\widetilde{f}\|_\infty \leq (2e)^k \|f\|_\infty$.*

Proof. The first statement is an immediate corollary of either Theorem 2.1 (taking $\Phi(u) = u^p$ and $p \to \infty$) or Theorem 2.2 (taking $t = \|f\|_\infty$). The second statement is immediate from Remark 2.1, with the better constant $k^k/k!$ in case f is homogeneous. In the general case, we use the fact that if $f^{=j}$ denotes the degree-j part of f, then $\|f^{=j}\|_\infty \leq 2^j \|f\|_\infty$; this is also proved by Kwapień [28, Lemma 2]. Then

$$\left\|\widetilde{f}\right\|_\infty = \left\|\sum_{j=0}^{k} \widetilde{f^{=j}}\right\|_\infty \leq \sum_{j=0}^{k} \left\|\widetilde{f^{=j}}\right\|_\infty \leq \sum_{j=0}^{k} (j^j/j!) \left\|f^{=j}\right\|_\infty \leq \sum_{j=0}^{k} (j^j/j!) 2^j \left\|f\right\|_\infty$$
$$\leq (2e)^k \|f\|_\infty. \qquad \blacktriangleleft$$

▶ Remark. Classical decoupling theory has not been too concerned with the dependence of constants on k, and most statements like Theorem 2.2 in the literature simply write $D_k = C_k$ to conserve symbols. However there are good reasons to retain the distinction, since making C_k small is usually much more important than making D_k small. For example, we can deduce Corollary 2.3 from Theorem 2.2 regardless of D_k's value.

Let us give an example application of these fundamental decoupling results. In a recent work comparing quantum query complexity to classical randomized query complexity, Aaronson and Ambainis [5] proved[2] the following:

▶ **Theorem 2.4.** *Let f be an N-variate degree-k homogeneous block-multilinear polynomial with real coefficients. Assume that under uniformly random ± 1 inputs we have $\|f\|_\infty \leq 1$. Then there is a randomized query algorithm making $2^{O(k)}(N/\epsilon^2)^{1-1/k}$ nonadaptive queries to the coordinates of $x \in \{\pm 1\}^N$ that outputs an approximation to $f(x)$ that is accurate to within $\pm\epsilon$ (with high probability).*

The authors "strongly conjecture[d]" that the assumption of block-multilinearity could be removed, and gave a somewhat lengthy proof of this conjecture in the case of $k = 2$, using [13] . We note that the full conjecture follows almost immediately from full decoupling:

▶ **Theorem 2.5.** *Aaronson and Ambainis's Theorem 2.4 holds without the assumption of block-multilinearity or homogeneity.*

Proof. Given a non-block-multilinear f on N variables ranging in $\{\pm 1\}$, consider its full decoupling $\widetilde{f}$ on kN variables. By Corollary 2.3 we have $\|\widetilde{f}\|_\infty \leq (2e)^k$. Let $g = (2e)^{-k}\widetilde{f}$, so that $g : \{\pm 1\}^{kN} \to [-1, +1]$ is a degree-k block-multilinear polynomial with $f(x) = (2e)^k g(x, x, \ldots, x)$. Now given query access to $x \in \{\pm 1\}^N$ and an error tolerance ϵ, we apply Theorem 2.4 to $g(x, x, \ldots, x)$ with error tolerance $\epsilon_1 = (2e)^{-k}\epsilon$; note that we can simulate queries to $(x, x, \ldots, x)$ using queries to x. This gives the desired query algorithm, and it makes $2^{O(k)}(kN/\epsilon_1^2)^{1-1/k} = 2^{O(k)}(N/\epsilon^2)^{1-1/k}$ queries as claimed. There is one more minor point: Theorem 2.4 requires its function to be homogeneous in addition to block-multilinear. However this assumption is easily removed by introducing k dummy variables treated as $+1$, and padding the monomials with them. ◀

2.2 Our one-block decoupling theorems, and the AA Conjecture

We now state our new versions of Theorems 2.1, 2.2 which apply only to one-block decoupling, but that have *polynomial* dependence of C_k on k. Proofs are deferred to Section 4.

As before, let $f(x) = \sum_{|S| \leq k} a_S x_S$ be an n-variate multivariate polynomial of degree at most k with coefficients a_S in a Banach space; let $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ consist of independent real random variables with all moments finite, and let $\boldsymbol{y}$, $\boldsymbol{z}$ be independent copies. We consider three slightly different hypotheses:

- **H1:** $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \sim \mathrm{N}(0, 1)$ are standard Gaussians.
- **H2:** $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ are uniformly random ± 1 values.
- **H3:** $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ are uniformly random ± 1 values and f is homogeneous.

[2] Actually, there is a small gap in their proof. In the line reading "By the concavity of the square root function…", they claim that $\|\boldsymbol{X}\|_1 \geq \|\boldsymbol{X}\|_2$ when $\boldsymbol{X}$ is a degree-k polynomial of uniformly random ± 1 bits. In fact the inequality goes the other way in general. But the desired inequality does hold up to a factor of e^k by [33, Theorem 9.22], and this is sufficient for their proof.

▶ **Theorem 2.6.** *If $\Phi : \mathbb{R}^{\geq 0} \to \mathbb{R}^{\geq 0}$ is convex and nondecreasing, then*

$$\mathbf{E}\Big[\Phi\Big(\Big\|\breve{f}(\boldsymbol{y},\boldsymbol{z})\Big\|\Big)\Big] \leq \mathbf{E}\Big[\Phi\Big(C_k \|f(\boldsymbol{x})\|\Big)\Big].$$

Also, if $t > 0$ (and we assume f's coefficients a_S are real under **H2**, **H3***), then*

$$\mathbf{Pr}\Big[\Big\|\breve{f}(\boldsymbol{y},\boldsymbol{z})\Big\| > C_k t\Big] \leq D_k \,\mathbf{Pr}\Big[\|f(\boldsymbol{x})\| > t\Big].$$

Here

$$C_k = \begin{cases} O(k) & \text{under } \mathbf{H1}, \\ O(k^2) & \text{under } \mathbf{H2}, \\ O(k^{3/2}) & \text{under } \mathbf{H3}, \end{cases} \qquad D_k = \begin{cases} O(k) & \text{under } \mathbf{H1}, \\ k^{O(k)} & \text{under } \mathbf{H2, H3}. \end{cases}$$

▶ Remark. It may seem that for the Φ-inequality in the Gaussian case, Kwapień's result mentioned in Remark 2.1 is better than ours, since he achieves full decoupling with a better constant than we get for one-block decoupling. But actually they are incomparable; the reason is the different scaling mentioned in Remark 1.1.

▶ Remark. As we will explain later in Remark 3.1, the bound $C_k = O(k)$ under **H1** is best possible (assuming that $D_k \leq \exp(O(k^2))$).

An immediate consequence of the above theorem, as in Corollary 2.3, is the following:

▶ **Corollary 2.7.** *If $f : \{\pm 1\}^n \to \mathbb{R}$ then $\|\breve{f}\|_\infty \leq O(k^2)\|f\|_\infty$.*

Let us now give an example of how one-block decoupling can be as useful as full decoupling, and why it is important to obtain $C_k = \mathrm{poly}(k)$. A very notable open problem in analysis of Boolean functions is the *Aaronson–Ambainis (AA) Conjecture*, originally proposed in 2008 [2, 4]:

AA Conjecture. *Let $f : \{\pm 1\}^n \to [-1, +1]$ be computable by a multilinear polynomial of degree at most k, $f(x) = \sum_{|S| \leq k} a_S x_S$. Then $\mathbf{MaxInf}_i[f] \geq \mathrm{poly}(\mathbf{Var}[f]/k)$.*

Here we use the standard notations for influences and variance:

$$\mathbf{MaxInf}_i[f] = \max_{i \in [n]} \{\mathbf{Inf}_i[f]\}, \quad \mathbf{Inf}_i[f] = \sum_{S \ni i} a_S^2, \qquad \mathbf{Var}[f] = \sum_{S \neq \emptyset} a_S^2, \qquad \|f\|_2^2 = \sum_S a_S^2.$$

The AA Conjecture is known to imply (and was directly motivated by) the following folklore conjecture concerning the limitations of quantum computation, dated to 1999 or before [4]:

Quantum Conjecture. *Any quantum query algorithm solving a Boolean decision problem using T queries can be correctly simulated on a $1 - \epsilon$ fraction of all inputs by a classical query algorithm using $\mathrm{poly}(T/\epsilon)$ queries.*

Because of their importance for quantum computation, Aaronson has twice listed these conjectures as "semi-grand challenges for quantum computing theory" [1, 3].

The best known result in the direction of the AA Conjecture [4] obtains an influence lower bound of $\mathrm{poly}(\mathbf{Var}[f])/\exp(O(k))$, using the DFKO Inequality [13]. Here we observe that there is a "one-line" deduction of this bound under the assumption that f is one-block

decoupled.[3] To see this, suppose that f is indeed one-block decoupled, so it can be written as $f(y,z) = \sum_{i=1}^{n} y_i g_i(z)$, where $g_i(z) = \sum_{S \ni i} a_S z_{S \setminus i}$ is the ith "derivative" of f. Observe that $\|g_i\|_2^2 = \mathbf{Inf}_i[f]$ and hence $\sum_{i=1}^{n} \|g_i\|_2^2 \geq \mathbf{Var}[f]$. Also note that for any $z \in \{\pm 1\}^n$ we must have $\sum_{i=1}^{n} |g_i(z)| \leq 1$, as otherwise we could achieve $|f(y,z)| > 1$ by choosing $y \in \{\pm 1\}^n$ appropriately. Taking expectations we get $\sum_{i=1}^{n} \|g_i\|_1 \leq 1$, and hence

$$e^{k-1} \geq e^{k-1} \sum_{i=1}^{n} \|g_i\|_1 \geq \sum_{i=1}^{n} \|g_i\|_2 \geq \frac{\sum_{i=1}^{n} \|g_i\|_2^2}{\max_{i=1}^{n} \|g_i\|_2} \geq \frac{\mathbf{Var}[f]}{\max_{i=1}^{n} \sqrt{\mathbf{Inf}_i[f]}}$$

$$\Rightarrow \qquad \mathbf{MaxInf}[f] \geq e^{2-2k} \mathbf{Var}[f]^2, \tag{1}$$

where the second inequality used the basic fact in analysis of Boolean functions [33, Theorem 9.22] that $\|g\|_2 \leq e^{k-1}\|g\|_1$ for $g : \{\pm 1\}^n \to \mathbb{R}$ of degree at most $k-1$.

The above gives a good illustration of how even one-block decoupling can already greatly simplify arguments in analysis of Boolean functions. We feel that (1) throws into sharp relief the challenge of improving $\exp(-O(k))$ to $1/\mathrm{poly}(k)$ for the AA Conjecture. We now use our results to show that the assumption that f is one-block decoupled is completely without loss of generality.

▶ **Theorem 2.8.** *The AA Conjecture holds if and only if it holds for one-block decoupled functions f.*

Proof. Suppose $f : \{\pm 1\}^n \to [-1,+1]$ has degree at most k. By Corollary 2.7 we get that $\|\breve{f}\|_\infty \leq C_k = O(k^2)$. Now $g = C_k^{-1} \breve{f}$ is one-block decoupled and has range $[-1,+1]$. Assuming the AA Conjecture holds for it, we get some $i \in [2n]$ such that $\mathbf{Inf}_i[g] \geq \mathrm{poly}(\mathbf{Var}[g]/k)$. Certainly this implies $\mathbf{Inf}_i[\breve{f}] \geq \mathrm{poly}(\mathbf{Var}[\breve{f}]/k)$. It is easy to see that $\mathbf{Inf}_i[f] = \mathbf{Inf}_i[\breve{f}]$ and $\mathbf{Inf}_i[f] \geq \mathbf{Inf}_{i+n}[\breve{f}]/(k-1)$ for all $i \in [n]$. Therefore letting $i' = \max\{i, i-n\} \in [n]$, we have $\mathbf{Inf}_{i'}[f] \geq \mathbf{Inf}_i[\breve{f}]/(k-1)$, and also $\mathbf{Var}[\breve{f}] \geq \mathbf{Var}[f]$. Thus $\mathbf{Inf}_{i'}[f] \geq \mathrm{poly}(\mathbf{Var}[f]/k)$, confirming the AA Conjecture for f. ◀

In particular, by combining this with (1) we recover the known $\mathrm{poly}(\mathbf{Var}[f])/\exp(O(k))$ lower bound for the AA Conjecture as applied to general f.

▶ Remark. Aaronson and Ambainis [5] recently observed that for the purposes of deriving the Quantum Conjecture, it suffices to prove the AA Conjecture for fully decoupled f. However the AA Conjecture is of significant interest in analysis of Boolean functions in and of itself, even independent of the Quantum Conjecture. Thus we feel Theorem 2.8 is worth knowing, especially in light of the simple argument (1).

3 Tight versions of the DFKO theorems

This section is concerned with analysis of Boolean functions $f : \{\pm 1\}^n \to \mathbb{R}$. We will use traditional Fourier notation, writing $f(x) = \sum_{S \subseteq [n]} \widehat{f}(S) x_S$. A key theme in this field is the dichotomy between functions with "Gaussian-like" behavior and functions that are essentially "juntas". Recall that f is said to be an (ϵ, C)-junta if $\|f - g\|_2^2 \leq \epsilon$ for some $g : \{\pm 1\}^n \to \mathbb{R}$ depending on at most C input coordinates. Partially exemplifying this theme is a family of theorems stating that any Boolean function f which is not essentially a junta must have a large "Fourier tail" – something like $\sum_{|S|>k} \widehat{f}(S)^2 > \delta$. Examples of such results include

[3] This observation is joint with John Wright.

Friedgut's Average Sensitivity Theorem [15], the FKN Theorem [17] (sharpened in [19, 33]), the Kindler–Safra Theorem [27, 25], and the Bourgain Fourier Tail Theorem [8]. The last of these implies that any $f : \{\pm 1\}^n \to \{\pm 1\}$ which is not a $(.01, k^{O(k)})$-junta must satisfy $\sum_{|S|>k} \widehat{f}(S)^2 > k^{-1/2+o(1)}$. This $k^{-1/2+o(1)}$ bound was made more explicit in [23], and the optimal bound of $\Omega(k^{-1/2})$ was obtained in [26]. These "Fourier tail" theorems have had application in fields such as PCPs and inapproximability [22, 12], sharp threshold theory [16], extremal combinatorics [14], and social choice [17].

All of the aforementioned theorems concern Boolean-*valued* functions; i.e., those with range $\{\pm 1\}$. By contrast, the DFKO Fourier Tail Theorem [13] is a result of this flavor for *bounded* functions; i.e., those with range $[-1, +1]$.

DFKO Fourier Tail Theorem. *Suppose $f : \{\pm 1\}^n \to [-1, +1]$ is not an $(\epsilon, 2^{O(k)}/\epsilon^2)$-junta. Then*

$$\sum_{|S|>k} \widehat{f}(S)^2 > \exp(-O(k^2 \log k)/\epsilon).$$

Most applications do not use this Fourier tail theorem directly. Rather, they use a key intermediate result, [13, Theorem 3], which we will refer to as the "DFKO Inequality". This was the case, for example, in a recent work on approximation algorithms for the Max-kXOR problem [6].

DFKO Inequality. *Suppose $f : \{\pm 1\}^n \to \mathbb{R}$ has degree at most k and $\mathbf{Var}[f] \geq 1$. Let $t \geq 1$ and suppose that $\mathbf{MaxInf}[f] \leq 2^{-O(k)}/t^2$. Then $\mathbf{Pr}[|f(\boldsymbol{x})| > t] \geq \exp(-O(t^2 k^2 \log k))$.*

Returning to the theme of "Gaussian-like behavior" versus "junta" behavior, we may add that the DFKO results straightforwardly imply (by the Central Limit Theorem) analogous, simpler-to-state results concerning functions on Gaussian space and Hermite tails. We record these generic consequences here; see, e.g., [33, Sections 11.1, 11.2] for a general discussion of such implications, and the definitions of Hermite coefficients $\widehat{f}(\alpha)$.

▶ **Corollary 3.1.** *Any $f : \mathbb{R}^n \to [-1, +1]$ satisfies the Hermite tail bound*

$$\sum_{|\alpha|>k} \widehat{f}(\alpha)^2 > \exp(-O(k^2 \log k)/\mathbf{Var}[f]).$$

Furthermore, suppose $\boldsymbol{z}$ is a standard n-dimensional Gaussian random vector and $t \geq 1$. Then any n-variate polynomial f of degree at most k with $\mathbf{Var}[f(\boldsymbol{z})] \geq 1$ satisfies $\mathbf{Pr}[|f(\boldsymbol{z})| > t] \geq \exp(-O(t^2 k^2 \log k))$.

Even though the Gaussian results in Corollary 3.1 are formally easier than their Boolean counterparts, we are not aware of any way to prove them – even in the case $n = 1$ – except via DFKO.

Tightness of the bounds. In [13, Section 6] it is shown that the results in Corollary 3.1 are tight, up to the $\log k$ factor in the exponent; this implies the same statement about the DFKO Fourier Tail Theorem and the DFKO Inequality. The tight example in both cases is essentially the univariate, degree-k Chebyshev polynomial.[4] In the next section we will show

[4] Formally speaking, [13, Section 6] only argues tightness of the Boolean theorems, but their constructions are directly based on the degree-k Chebyshev polynomial applied to a single standard Gaussian.

how to use our one-block decoupling result to remove the $\log k$ in the exponential from both DFKO theorems. The results immediately transfer to the Gaussian setting, and we therefore obtain the tight $\exp(-\Theta(k^2))$ bound for all versions of the inequality.

Our method of proof is actually to *first* prove the results in the Gaussian setting, where the one-block decoupling makes the proofs quite easy. Then we can transfer the results to the Boolean setting by using the Invariance Principle [32]. This methodology – proving the more natural Gaussian tail bound first, then transferring the result to the Boolean setting via Invariance – is quite reminiscent of how the optimal form of Bourgain's Fourier Tail Theorem was recently obtained [26].

There is actually an additional, perhaps unexpected, bonus of our proof methodology; we show that the bound in the DFKO Inequality can be improved from $\exp(-O(t^2k^2))$ to $\exp(-O(t^2k))$ whenever f is *homogeneous*.

3.1 Proofs of the tight DFKO theorems

We begin with a tail-probability lower bound for one-block decoupled polynomials of Gaussians.

▶ **Lemma 3.2.** *Suppose $f(y,z) = \sum_{i=1}^n y_i g_i(z)$ is a one-block decoupled polynomial on $n+n$ variables, with real coefficients and degree at most k. Let $\boldsymbol{y}, \boldsymbol{z} \in \mathrm{N}(0,1)^n$ be independent standard n-dimensional Gaussians and write*

$$\sigma^2 = \mathbf{Var}[f(\boldsymbol{y},\boldsymbol{z})] = \sum_{i=1}^n \|g_i\|_2^2. \tag{2}$$

Then for $u > 0$ we have $\mathbf{Pr}[|f(\boldsymbol{y},\boldsymbol{z})| > u] \geq \exp(-O(k + u^2/\sigma^2))$.

Proof. Let $v(z) = \sum_{i=1}^n g_i(z)^2$, a polynomial of degree at most $2(k-1)$ in $z_1, \dots, z_n$. By (2) we have $\mathbf{E}[v(\boldsymbol{z})] = \sigma^2$. We now use Theorem 1.3 to get

$$\mathbf{Pr}[v(\boldsymbol{z}) > \sigma^2] \geq \frac{1}{4} e^{-2(2k-1)} = \exp(-O(k)).$$

On the other hand, for any outcome $\boldsymbol{z} = z$ we have that $f(\boldsymbol{y}, z) \sim \mathrm{N}(0, v(z))$. Thus

$$v(z) > \sigma^2 \quad \Longrightarrow \quad \mathbf{Pr}[|f(\boldsymbol{y},z)| > u] \geq \Omega(e^{-u^2/2\sigma^2}).$$

Combining the previous two statements completes the proof, since $\boldsymbol{y}$ and $\boldsymbol{z}$ are independent. ◀

We can now prove an optimal version of the DFKO Inequality in the Gaussian setting. It is also significantly better in the homogeneous case.

▶ **Theorem 3.3.** *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a polynomial of degree at most k, and let $\boldsymbol{x} \sim \mathrm{N}(0,1)^n$ be a standard n-dimensional Gaussian vector. Assume $\mathbf{Var}[f(\boldsymbol{x})] \geq 1$. Then for $t \geq 1$ it holds that $\mathbf{Pr}[|f(\boldsymbol{x})| > t] \geq \exp(-O(t^2k^2))$. Furthermore, if f is multilinear and homogeneous then the lower bound may be improved to $\exp(-O(t^2k))$.*

Proof. For any n-variate polynomial of Gaussians, we can find an N-variate multilinear polynomial of Gaussians of no higher degree that is arbitrarily close in Lévy distance (see, e.g., [20, Lemma 15], or use the CLT to pass to ± 1 random variables, then Invariance to pass back to Gaussians). Note, however, that this transformation does not preserve homogeneity. In any case, we can henceforth assume f is multilinear, $f(x) = \sum_{|S| \leq k} a_S x_S$.

For independent $\boldsymbol{y}, \boldsymbol{z} \sim \mathrm{N}(0,1)^n$, observe that

$$\mathbf{Var}[\breve{f}(\boldsymbol{y},\boldsymbol{z})] = \sum_{j=1}^{k} j \sum_{|S|=j} a_S^2 \geq \sum_{S \neq \emptyset} a_S^2 = \mathbf{Var}[f(\boldsymbol{x})] \geq 1,$$

and if f is homogeneous we get the better bound $\mathbf{Var}[\breve{f}(\boldsymbol{y},\boldsymbol{z})] \geq k$. By our Theorem 2.6 on one-block decoupling, we have

$$\mathbf{Pr}\Big[\Big|f(\boldsymbol{x})\Big| > t\Big] \geq D_k^{-1}\,\mathbf{Pr}\Big[\Big|\breve{f}(\boldsymbol{y},\boldsymbol{z})\Big| > C_k t\Big],$$

where $C_k = D_k = O(k)$. The theorem is now an immediate consequence of Lemma 3.2. ◀

▶ Remark 3.1. A consequence of this proof is that – assuming $D_k \leq \exp(O(k^2))$ – it is impossible to asymptotically improve on our $C_k = O(k)$ in Theorem 2.6 in the Gaussian setting **H1**. Otherwise, we would achieve a bound of $\exp(-o(k^2))$ in Theorem 3.3, contrary to the example in [13, Section 6].

We can now obtain the sharp DFKO Inequality in the Boolean setting by using the Invariance Principle.

▶ **Corollary 3.4.** *Theorem 3.3 holds when $\boldsymbol{x} \sim \{\pm 1\}^n$ is uniform and we additionally assume that $\mathbf{MaxInf}[f] \leq \exp(-Ct^2k^2)$, or just $\exp(-Ct^2k)$ in the homogeneous case. Here C is a universal constant.*

Proof. This follows immediately from the Lévy distance bound in [32, Theorem 3.19, Hypothesis 4]. We only need to ensure that the Lévy distance is noticeably less than the target lower bound we're aiming for. (We also remark that the Invariance Principle transformation preserves variance and homogeneity.) ◀

Next, we obtain the sharp DFKO Fourier Tail Theorem. Its deduction from the DFKO Inequality in [13] is unfortunately not "black-box", so we will have to give a proof.

▶ **Corollary 3.5.** *Suppose $f : \{\pm 1\}^n \to [-1,+1]$ is not an $(\epsilon, 2^{O(k^2/\epsilon)})$-junta. Then*

$$\sum_{|S|>k} \widehat{f}(S)^2 > \exp(-O(k^2)/\epsilon). \tag{3}$$

Proof. We use notation and basic results from [33]. Given $f : \{\pm 1\}^n \to [-1,+1]$, let $J = \{i \in [n] : \mathbf{Inf}_i^{\leq k}[f] > \exp(-Ak^2/\epsilon)\}$, where A is a large constant to be chosen later. Since $\|f\|_2^2 \leq 1$ it follows easily that $|J| \leq 2^{O(k^2/\epsilon)}$. Now define $g = f - f^{\subseteq J}$; note that g has range in $[-2,+2]$ since $f^{\subseteq J}$ has range in $[-1,+1]$, being an average of f over the coordinates outside J. If $\|g\|_2^2 < \epsilon/2$ then f is $\epsilon/2$-close to the $2^{O(k^2/\epsilon)}$-junta $f^{\subseteq J}$ and we are done. Otherwise, $\|g\|_2^2 \geq \epsilon/2$ and we let $h = g^{\leq k}$. If $\|h-g\|_2^2 > \epsilon/4$ then we immediately conclude that $\sum_{|S|>k} \widehat{f}(S)^2 > \epsilon/4$, which is more than enough to be done. Otherwise $\|h-g\|_2^2 \leq \epsilon/4$, from which we conclude $\|h\|_2^2 \geq \epsilon/4$. Now h has degree at most k and satisfies $\mathbf{Inf}_i[h] \leq \exp(-Ak^2/\epsilon)$ for all $i \notin J$. Let $\widetilde{h}$ denote the mixed Boolean/Gaussian function which has the same multilinear form as h, but where we think of the coordinates in J as being ± 1 random variables and the coordinates not in J as being standard Gaussians. We now "partially" apply the Invariance Principle [32, Theorem 3.19] to h, in the sense that we only hybridize over the coordinates not in J. We conclude that the Lévy distance between h and $\widetilde{h}$ is at most $\exp(-\Omega(Ak^2/\epsilon))$. Our goal is now to show that

$$\mathbf{Pr}[|\widetilde{h}| > 3] \geq \exp(-O(k^2/\epsilon)), \tag{4}$$

where the constant in the $O(\cdot)$ does not depend on A. Having shown this, by taking A large enough the Lévy distance bound lets us deduce (4) for h as well. But then since $|g| \le 2$ always, we may immediately deduce $\|g-h\|_2^2 \ge \exp(-O(k^2)/\epsilon)$ and hence (3).

It remains to verify (4). For each restriction x_J to the J-coordinates, the function $\widetilde{h}_{x_J}$ is a multilinear polynomial in independent Gaussians with some variance $\sigma_{x_J}^2$. From Theorem 3.3 we can conclude that $\mathbf{Pr}[|\widetilde{h}_{x_J}| > 3] \ge \exp(-O(k^2/\sigma_{x_J}^2))$. Thus if we can show $\sigma_{\boldsymbol{x}_J}^2 \ge \Omega(\epsilon)$ with probability at least $2^{-O(k)}$ when $\boldsymbol{x}_J \in \{\pm 1\}^J$ is uniformly random, we will have established (4). But this follows similarly as in Lemma 3.2. Note that $\sigma_{x_J}^2 = \mathbf{E}[\widetilde{h}_{x_J}^2]$, since h has no constant term. Now $\sigma_{x_J}^2$ is a degree-$2k$ polynomial in x_J, and its expectation is simply $\|h\|_2^2 \ge \epsilon/4$, so Theorem 1.3 indeed implies that $\mathbf{Pr}[\sigma_{\boldsymbol{x}_J}^2 \ge \epsilon/4] \ge 2^{-O(k)}$ and we are done. ◀

▶ Remark. We comment that the dependence of $\mathbf{MaxInf}[f]$ on t in Corollary 3.4, and the junta size in Corollary 3.5, are not as good as in [13]. This seems to be a byproduct of the use of Invariance.

A similar (but easier) proof can be used to derive the following Gaussian version of Corollary 3.5; alternatively, one can use a generic CLT argument, noting that the only "junta" a Gaussian function can be close to is a constant function:

▶ **Corollary 3.6.** *Any $f : \mathbb{R}^n \to [-1,+1]$ satisfies the Hermite tail bound*

$$\sum_{|\alpha|>k} \widehat{f}(\alpha)^2 > \exp(-O(k^2)/\mathbf{Var}[f]).$$

This strictly improves upon Corollary 3.1.

4 Proofs of our one-block decoupling theorems

In this section we prove Theorem 2.6. The key idea of the proof is to express $\breve{f}(y,z)$ as a "small" linear combination of expressions of the form $f(\alpha_i x + \beta_i y)$, where $\alpha_i^2 + \beta_i^2 = 1$ (in the Gaussian case) or $|\alpha_i| + |\beta_i| = 1$ (in the Boolean case). The following is the central lemma.

▶ **Lemma 4.1.** *In the setting of Theorem 2.6, there exists $m = O(k)$ and $\alpha, \beta, c \in \mathbb{R}^m$ such that*

- $\breve{f}(y,z) = \sum_{i=1}^m c_i f(\alpha_i y + \beta_i z)$;
- $\sum_{i=1}^m |c_i| \le C_k$;
- $\alpha_i^2 + \beta_i^2 = 1$ *for all* $i \in [m]$ *under* **H1**, *and* $|\alpha_i| + |\beta_i| = 1$ *for all* $i \in [m]$ *under* **H2**, **H3**;
- $|\alpha_i|, |\beta_i| \ge 1/O(C_k)$ *for all* $i \in [m]$.

With Lemma 4.1 in hand, the proof of Theorem 2.6 is quite straightforward in the Gaussian case, and not much more difficult in the Boolean case. We show these deductions first.

Proof of Theorem 2.6 under Hypothesis H1. By Lemma 4.1, for any convex nondecreasing function $\Phi : \mathbb{R}^{\ge 0} \to \mathbb{R}^{\ge 0}$ we have

$$\begin{aligned}\mathbf{E}\Big[\Phi\Big(\Big\|\breve{f}(\boldsymbol{y},\boldsymbol{z})\Big\|\Big)\Big] &= \mathbf{E}\Big[\Phi\Big(\Big\|\sum_{i=1}^m c_i f(\alpha_i \boldsymbol{y} + \beta_i \boldsymbol{z})\Big\|\Big)\Big] \\ &\le \mathbf{E}\Big[\Phi\Big(\sum_{i=1}^m |c_i| \Big\| f(\alpha_i \boldsymbol{y} + \beta_i \boldsymbol{z})\Big\|\Big)\Big]\end{aligned}$$

$$\begin{aligned}
&\leq \sum_{i=1}^{m} \frac{|c_i|}{C_k} \mathbf{E}\left[\Phi\left(C_k \|f(\alpha_i \boldsymbol{y} + \beta_i \boldsymbol{z})\|\right)\right] \\
&= \sum_{i=1}^{m} \frac{|c_i|}{C_k} \mathbf{E}[\Phi(C_k \|f(\boldsymbol{x})\|)] \\
&\leq \mathbf{E}[\Phi(C_k \|f(\boldsymbol{x})\|].
\end{aligned}$$

Here the inequalities follow from the convexity and monotonicity of Φ, and the second equality holds because $\alpha_i \boldsymbol{y} + \beta \boldsymbol{z} \sim \mathrm{N}(0,1)^n$ due to $\alpha_i^2 + \beta_i^2 = 1$.

As for the tail-bound comparison, by Lemma 4.1, whenever y, z are such that $\|\breve{f}(y,z)\| > C_k t$, the triangle inequality implies that there must exist at least one $i \in [m]$ with $\|f(\alpha_i y + \beta_i z)\| > t$. It follows that there must exist at least one $i \in [m]$ such that

$$\mathbf{Pr}[\|f(\alpha_i \boldsymbol{y} + \beta_i \boldsymbol{z})\| > t] \geq \frac{1}{m} \mathbf{Pr}[\|\breve{f}(\boldsymbol{y}, \boldsymbol{z})\| > C_k t].$$

This completes the proof, since $\alpha_i \boldsymbol{y} + \beta_i \boldsymbol{z} \sim \mathrm{N}(0,1)^n$ and $m = O(k)$. ◀

Proof of Theorem 2.6 under Hypotheses H2, H3. We define ± 1 random variables as follows:

$$\boldsymbol{x}_j^{(i)} = \begin{cases} \mathrm{sgn}(\alpha_i)\boldsymbol{y}_j & \text{with probability } |\alpha_i|, \\ \mathrm{sgn}(\beta_i)\boldsymbol{z}_j & \text{with probability } |\beta_i|, \end{cases}$$

for all $i \in [m]$ and $j \in [n]$ independently. Notice that each $\boldsymbol{x}^{(i)}$ is distributed uniformly on $\{\pm 1\}^n$, though they are not independent. To prove the desired inequality concerning Φ, we can repeat the proof in the Gaussian case, except that we no longer have the identity

$$\mathbf{E}\left[\Phi\left(C_k \|f(\alpha_i \boldsymbol{y} + \beta_i \boldsymbol{z})\|\right)\right] = \mathbf{E}[\Phi(C_k \|f(\boldsymbol{x})\|)].$$

In fact we will show that the left-hand side is at most the right-hand side. Notice that for all fixed $y, z \in \{\pm 1\}^n$, the multilinearity of f implies that

$$f(\alpha_i y + \beta_i z) = \mathbf{E}[f(\boldsymbol{x}^{(i)}) \mid (\boldsymbol{y}, \boldsymbol{z}) = (y, z)]. \tag{5}$$

Thus

$$\begin{aligned}
\mathbf{E}\left[\Phi\left(C_k \|f(\alpha_i \boldsymbol{y} + \beta_i \boldsymbol{z})\|\right)\right] &= \mathop{\mathbf{E}}_{\boldsymbol{y},\boldsymbol{z}}\left[\Phi\left(C_k \left\| \mathop{\mathbf{E}}_{\boldsymbol{x}^{(i)}|\boldsymbol{y},\boldsymbol{z}}\left[f(\boldsymbol{x}^{(i)})\right]\right\|\right)\right] \\
&\leq \mathop{\mathbf{E}}_{\boldsymbol{y},\boldsymbol{z}} \mathop{\mathbf{E}}_{\boldsymbol{x}^{(i)}}\left[\Phi\left(C_k \|f(\boldsymbol{x}^{(i)})\|\right)\right] = \mathbf{E}\left[\Phi\left(C_k \|f(\boldsymbol{x})\|\right)\right],
\end{aligned}$$

as claimed, where we used convexity again.

As for the tail-bound comparison, recall that we are now assuming f has real coefficients. As in the Gaussian case there is at least one $i \in [m]$ with

$$\mathbf{Pr}[|f(\alpha_i \boldsymbol{y} + \beta_i \boldsymbol{z})| > t] \geq \frac{1}{O(k)} \mathbf{Pr}[|\breve{f}(\boldsymbol{y}, \boldsymbol{z})| > C_k t].$$

Now suppose y, z are such that $|f(\alpha_i y + \beta_i z)| > t$ and consider the conditional distribution on $\boldsymbol{x}^{(i)}$. If we can show that, conditionally, $\mathbf{Pr}[|f(\boldsymbol{x}^{(i)})| > t] \geq k^{-O(k)}$ then we are done. But from (5) we have that $\left|\mathbf{E}[f(\boldsymbol{x}^{(i)})]\right| > t$; therefore the desired result follows from Theorem 1.3 and the fact that $\min(|\alpha_i|, |\beta_i|) \geq 1/O(C_k) = 1/\mathrm{poly}(k)$. ◀

4.1 Proof of Lemma 4.1

The proof of Lemma 4.1 involves minimizing $\sum_{i=1}^{m} |c_i|$ by carefully setting the ratios of α_i and β_i to be a hyperharmonic progression.

Proof of Lemma 4.1. The main work involves treating the homogeneous case.

Homogeneous case. Our goal for homogeneous f is to write

$$\breve{f}(y,z) = \sum_{i=1}^{k+1} c_i f(\alpha_i y + \beta_i z).$$

Comparing the expressions term by term, it is equivalent to say that for any $S \subseteq [n]$ with $|S| = k$,

$$\sum_{j \in S} y_j z_{S/j} = \sum_{i=1}^{k+1} c_i \prod_{j \in S} (\alpha_i y_j + \beta_i z_j).$$

We can further simplify this to the conditions

$$\sum_{i=1}^{k+1} c_i \alpha_i^{k-t} \beta_i^t = \begin{cases} 1 & \text{if } t = k-1 \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

for all integers $0 \le t \le k$. Let us write $\Delta_i = \frac{\beta_i}{\alpha_i}$ and introduce the Vandermonde matrix

$$V = \begin{bmatrix} 1 & 1 & \dots & 1 \\ \Delta_1 & \Delta_2 & \cdots & \Delta_{k+1} \\ \cdots & \cdots & \cdots & \cdots \\ \Delta_1^{k-1} & \Delta_2^{k-1} & \dots & \Delta_{k+1}^{k-1} \\ \Delta_1^{k} & \Delta_2^{k} & \cdots & \Delta_{k+1}^{k} \end{bmatrix}.$$

We will also write A for the diagonal matrix $\mathrm{diag}(\alpha_1^k, \alpha_2^k, \dots, \alpha_{k+1}^k)$, and write e_k for the indicator vector of the kth coordinate, $e_k = (0, 0, \dots, 0, 1, 0)$. Then the necessary conditions (6) are equivalent to the matrix equation $VAc = e_k$. Assuming all the Δ_i's are different, V is invertible and there is an explicit formula for its inverse [30]. This yields the following expression for the $c_1, \dots, c_{k+1}$ in terms of α and β:

$$c_i = (A^{-1}V^{-1}e_k)_i = \frac{1}{\alpha_i^k} \cdot \frac{\Delta_i - \sum_{j=1}^{k+1} \Delta_j}{\prod_{j=1, j\neq i}^{k+1} (\Delta_i - \Delta_j)}. \tag{7}$$

The main illustrative case: Hypothesis H1 and k odd. We will now assume that k is odd; this assumption will be easily removed later. It will henceforth be convenient to replace our indices $1, \dots, k+1$ with the following slightly peculiar but symmetric set of indices:

$$I = \left\{\pm 1, \pm 2, \dots, \pm \tfrac{k-1}{2}, \pm \tfrac{1}{2}\right\}.$$

Now under Hypothesis **H1**, we will choose

$$\alpha_i = \frac{i}{\sqrt{k^2 + i^2}}, \quad \beta_i = \frac{k}{\sqrt{k^2 + i^2}} \quad \Longrightarrow \quad \Delta_i = \frac{k}{i}$$

for all $i \in I$. These choices satisfy $\alpha_i^2 + \beta_i^2 = 1$ and $|\alpha_i|, |\beta_i| \geq 1/O(C_k)$, so it remains to prove that for c defined by (7) we have $\sum |c_i| \leq O(k)$.

Let us upper-bound all $|c_i|$. Since it easy to see that $|c_i| = |c_{-i}|$ for all $i \in I$, it will suffice for us to consider the positive $i \in I$. For $1 \leq i \leq \frac{k-1}{2}$, we have

$$\begin{aligned}
\left| \prod_{j \in I, j \neq i} (\Delta_i - \Delta_j) \right| &= (\Delta_{1/2} - \Delta_i)(\Delta_i - \Delta_{-1/2}) \cdot \prod_{j=1, j\neq i}^{(k-1)/2} |\Delta_i - \Delta_j| \cdot \prod_{j=-(k-1)/2}^{-1} (\Delta_i - \Delta_j) \\
&= \left(2k - \frac{k}{i}\right)\left(2k + \frac{k}{i}\right) \cdot \prod_{j=1, j\neq i}^{(k-1)/2} \left|\frac{k}{i} - \frac{k}{j}\right| \cdot \prod_{j=1}^{(k-1)/2} \left(\frac{k}{i} + \frac{k}{j}\right) \\
&= k^k \left(4 - \frac{1}{i^2}\right) \cdot \prod_{j=1, j\neq i}^{(k-1)/2} \frac{|j-i|}{ij} \cdot \prod_{j=1}^{(k-1)/2} \frac{j+i}{ij} \\
&= \frac{k^k}{i^{k-2}} \left(4 - \frac{1}{i^2}\right) \frac{\left(\frac{k-1}{2} + i\right)! \left(\frac{k-1}{2} - i\right)!}{\left(\frac{k-1}{2}\right)!^2}.
\end{aligned}$$

Thus from (7),

$$\begin{aligned}
|c_i| &= \left(\frac{\sqrt{k^2 + i^2}}{i}\right)^k \cdot \frac{k}{i} \cdot \frac{i^{k-2}}{k^k} \cdot \frac{1}{4 - 1/i^2} \cdot \frac{\left(\frac{k-1}{2}\right)!^2}{\left(\frac{k-1}{2} + i\right)! \left(\frac{k-1}{2} - i\right)!} \\
&= \frac{k}{i^3} \left(1 + \frac{i^2}{k^2}\right)^{k/2} \frac{1}{4 - 1/i^2} \frac{\left(\frac{k-1}{2}\right)!^2}{\left(\frac{k-1}{2} + i\right)! \left(\frac{k-1}{2} - i\right)!}.
\end{aligned}$$

When $1 \leq i \leq \sqrt{k}$, we have

$$|c_i| = \frac{k}{i^3} \left(1 + \frac{i^2}{k^2}\right)^{k/2} \frac{1}{4 - 1/i^2} \frac{\left(\frac{k-1}{2}\right)!^2}{\left(\frac{k-1}{2} + i\right)! \left(\frac{k-1}{2} - i\right)!} \leq \frac{k}{i^3} \left(1 + \frac{1}{k}\right)^{k/2} \leq \frac{\sqrt{e}k}{i^3}.$$

For $\sqrt{k} \leq i \leq \frac{k-1}{2}$, consider the ratio between $(i+1)^3|c_{i+1}|$ and $i^3|c_i|$; it satisfies

$$\begin{aligned}
\frac{(i+1)^3|c_{i+1}|}{i^3|c_i|} &\leq \frac{(k^2 + (i+1)^2)^{k/2}}{(k^2 + i^2)^{k/2}} \cdot \frac{\frac{k-1}{2} - i}{\frac{k-1}{2} + i + 1} \\
&= \left(1 + \frac{2i+1}{k^2 + i^2}\right)^{k/2} \cdot \frac{k - 1 - 2i}{k + 1 + 2i} \\
&\leq \left(1 + \frac{2i+1}{k^2}\right)^{k/2} \cdot \frac{k - 1 - 2i}{k} \\
&\leq e^{\frac{2i+1}{2k}} \left(1 - \frac{2i+1}{k}\right) \leq 1.
\end{aligned}$$

The last inequality holds since $e^{x/2}(1-x) \leq 1$ for all $0 \leq x \leq 1$. Thus we have $(i+1)^3|c_{i+1}| \leq i^3|c_i|$, and hence by induction that

$$|c_i| \leq \frac{\sqrt{e}k}{i^3} \quad \forall\ 1 \leq i \leq \tfrac{k-1}{2}. \tag{8}$$

We now need to bound $c_{1/2}$. Similarly to the above, we have

$$\begin{aligned}\left|\prod_{j\in I, j\neq\frac{1}{2}}(\Delta_{1/2}-\Delta_j)\right| &= (\Delta_{\frac{1}{2}}-\Delta_{-1/2})\cdot\prod_{j=1}^{(k-1)/2}(\Delta_{1/2}-\Delta_j)\cdot\prod_{j=-(k-1)/2}^{-1}(\Delta_{\frac{1}{2}}-\Delta_j)\\ &= 4k\cdot\prod_{j=1}^{(k-1)/2}\left(2k-\frac{k}{j}\right)\cdot\prod_{j=1}^{(k-1)/2}(2k+\frac{k}{j})\\ &= 4k^k\cdot\prod_{j=1}^{(k-1)/2}\frac{2j-1}{j}\cdot\prod_{j=1}^{(k-1)/2}\frac{2j+1}{j}\\ &= 4k^k\frac{(k-2)!!k!!}{\left(\frac{k-1}{2}\right)!^2}\end{aligned}$$

Thus from (7) we get

$$\begin{aligned}|c_{1/2}| &= \frac{(\sqrt{k^2+(1/2)^2})^k}{(1/2)^k}\cdot 2k\cdot\frac{1}{4k^k}\cdot\frac{\left(\frac{k-1}{2}\right)!^2}{(k-2)!!k!!}\\ &= \left(1+\frac{1}{4k^2}\right)^{k/2}\left(\frac{(k-1)!!}{(k-2)!!}\right)^2 \le 4k. \qquad (9)\end{aligned}$$

Now combining (8), (9), we obtain

$$\sum_i |c_i| = 2\sum_{i=1}^{(k-1)/2}|c_i| + 2|c_{1/2}| \le 2\sqrt{e}\sum_{i=1}^{(k-1)/2}\frac{k}{i^3} + 8k \le 20k,$$

as needed.

Handling even k. If k is even, we define our index set to be

$$I = \left\{0, \pm1, \pm2, \ldots, \pm\tfrac{k-2}{2}, \pm\tfrac{1}{2}\right\}.$$

For $i \in I\setminus\{0\}$ we define α_i and β_i as before; we also define $\alpha_0 = 1$, $\beta_0 = 0$, and hence $\Delta_0 = 0$. It is easy to check that $c_0 = 0$ (and hence we haven't actually violated $|\beta_i| \ge 1/O(C_k)$), and the upper bounds for the other $|c_i|$ still hold. This completes the proof of the homogeneous case under Hypothesis **H1**.

Hypotheses H3. We explain the case of k odd; the same trick as before can be used for even k. For Hypothesis **H3** we use

$$\alpha_i = \frac{i}{k^{3/2}+|i|}, \quad \beta_i = \frac{k^{3/2}}{k^{3/2}+|i|} \quad\Longrightarrow\quad \Delta_i = \frac{k^{3/2}}{i},$$

which satisfy $|\alpha_i|+|\beta_i| = 1$ and $|\alpha_i|, |\beta_i| \ge 1/O(k^{3/2})$. Analysis similar to before shows that $\sum_i |c_i| \le O(k^{3/2})$. This completely finishes the proof under Hypothesis **H3**.

Hypothesis H2, the homogeneous case. Here we do something slightly different. For even or odd k we let the index set be $I = \{1, 2, \ldots, k, \frac{1}{2}\}$ and then define

$$\alpha_i = \frac{i^2}{k^2+i^2}, \quad \beta_i = \frac{k^2}{k^2+i^2} \quad\Longrightarrow\quad \Delta_i = \frac{k^2}{i^2}.$$

Now we have $|\alpha_i|+|\beta_i| = \alpha_i+\beta_i = 1$ and $|\alpha_i|, |\beta_i| \ge 1/O(k^2)$. Again, similar analysis shows that $\sum_i |c_i| \le O(k^2)$.

Extending to the non-homogeneous case under H2. Now we need to be concerned with the terms at degree $k' < k$. Here a key observation is that, since $\alpha_i + \beta_i = 1$ for all i, the following holds for all $k' < k$:

$$\sum_i c_i \alpha_i^{k'-t} \beta_i^t = \sum_i c_i \alpha_i^{k'-t} \beta_i^t (\alpha_i + \beta_i) = \sum_i c_i \alpha_i^{k'-t+1} \beta_i^t + \sum_i c_i \alpha_i^{k'-t} \beta_i^{t+1}.$$

Thus an induction shows that in fact

$$\sum_i c_i \alpha_i^{k'-t} \beta_i^t = \begin{cases} k - k' & \text{if } t = k' \\ 1 & \text{if } t = k' - 1 \\ 0 & \text{otherwise} \end{cases}$$

for all $k' \le k$. This is almost exactly what we need to treat the non-homogeneous case using all the same choices for c, α, β, except for the $t = k'$ case. But we can use a simple trick to fix this:

$$\frac{1}{2} \sum_i c_i \alpha_i^{k'-t} \beta_i^t - \frac{1}{2} \sum_i c_i (-\alpha_i)^{k'-t} \beta_i^t = \frac{1 - (-1)^{k'-t}}{2} \sum_i c_i \alpha_i^{k'-t} \beta_i^t = \begin{cases} 1 & \text{if } t = k' - 1 \\ 0 & \text{otherwise} \end{cases}$$

From this we get

$$\breve{f}(y, z) = \sum_{i=1}^{m} c_i f(\alpha_i y + \beta_i z)$$

even in the non-homogeneous case, with all the desired conditions and $m = 2(k + 1)$.

Extending to the non-homogeneous case under H1. The trick here for handling degree $k' < k$ is similar. Using the fact that $\alpha_i^2 + \beta_i^2 = 1$ for all i, we get that for all $k' < k$,

$$\sum_i c_i \alpha_i^{k'-t} \beta_i^t = \sum_i c_i \alpha_i^{k'-t} \beta_i^t (\alpha_i^2 + \beta_i^2) = \sum_i c_i \alpha_i^{k'-t+2} \beta_i^t + \sum_i c_i \alpha_i^{k'-t} \beta_i^{t+2}.$$

Then by induction, the we conclude that

$$\sum_{i=1}^{k+1} c_i \alpha_i^{k'-t} \beta_i^t = \begin{cases} 1 & \text{if } t = k' - 1 \\ 0 & \text{otherwise} \end{cases}$$

holds for all $0 \le k' \le k$ such that $k - k'$ is even. We are therefore almost done: we have established the **H1** case of Lemma 4.1 for all polynomials with only odd-degree terms or only even-degree terms. Finally, for a general polynomial f we can decompose it as $f = f_{\text{odd}} + f_{\text{even}}$, where f_{odd} (respectively, f_{even}) contains all the terms in f with odd (respectively, even) degree. We know that there exist some vectors α, β, c and α', β', c' satisfying

$$\breve{f}_{\text{odd}}(y, z) = \sum_{i=1}^{k+1} c_i f_{\text{odd}}(\alpha_i y + \beta_i z), \quad \breve{f}_{\text{even}}(y, z) = \sum_{i=1}^{k+1} c'_i f_{\text{even}}(\alpha'_i y + \beta'_i z),$$

and $\sum_i |c_i|, \sum_i |c'_i| \le 20k$. Thus

$$\begin{aligned}
\breve{f}(y,z) &= \breve{f}_{\text{odd}}(y,z) + \breve{f}_{\text{even}}(y,z) \\
&= \sum_{i=1}^{k+1} c_i f_{\text{odd}}(\alpha_i y + \beta_i z) + \sum_{i=1}^{k+1} c'_i f_{\text{even}}(\alpha'_i y + \beta'_i z) \\
&= \sum_{i=1}^{k+1} \frac{1}{2} c_i (f(\alpha_i y + \beta_i z) - f(-\alpha_i y - \beta_i z)) + \sum_{i=1}^{k+1} \frac{1}{2} c'_i (f(\alpha'_i y + \beta'_i z) + f(-\alpha'_i y - \beta'_i z)) \\
&= \sum_{i=1}^{4(k+1)} c''_i f(\alpha''_i y + \beta''_i z),
\end{aligned}$$

where $c'' = (c/2, -c/2, c'/2, c'/2), \alpha'' = (\alpha, -\alpha, \alpha', -\alpha'), \beta'' = (\beta, -\beta, \beta', -\beta')$ and $\sum_i |c''_i| \le 40k$. ◀

Acknowledgments. The authors would like to thank Oded Regev for helpful discussions, and John Wright for permission to include (1).

References

1. Scott Aaronson. Ten semi-grand challenges for quantum computing theory, 2005. `http://www.scottaaronson.com/writings/qchallenge.html`.
2. Scott Aaronson. How to solve longstanding open problems in quantum computing using only Fourier Analysis. Lecture at Banff International Research Station, 2008. `http://www.scottaaronson.com/talks/openqc.ppt`.
3. Scott Aaronson. Updated version of "ten semi-grand challenges for quantum computing theory", 2010. `http://www.scottaaronson.com/blog/?p=471`.
4. Scott Aaronson and Andris Ambainis. The need for structure in quantum speedups. *Theory Of Computing*, 10(6):133–166, 2014.
5. Scott Aaronson and Andris Ambainis. Forrelation: a problem that optimally separates quantum from classical computing. In *Proceedings of the 47th Annual ACM Symposium on Theory of Computing*, pages 307–316, 2015.
6. Boaz Barak, Ankur Moitra, Ryan O'Donnell, Prasad Raghavendra, Oded Regev, David Steurer, Luca Trevisan, Aravindan Vijayaraghavan, David Witmer, and John Wright. Beating the random assignment on constraint satisfaction problems of bounded degree. In *Proceedings of the 18th Annual International Workshop on Approximation Algorithms for Combinatorial Optimization Problems*, 2015.
7. Franck Barthe and Emanuel Milman. Transference principles for log-sobolev and spectral-gap with applications to conservative spin systems. *Communications in Mathematical Physics*, 323(2):575–625, 2013.
8. Jean Bourgain. On the distribution of the Fourier spectrum of Boolean functions. *Israel Journal of Mathematics*, 131(1):269–276, 2002.
9. Victor de la Peña. Decoupling and Khintchine's inequalities for U-statistics. *Annals of Probability*, 20(4):1877–1892, 1992.
10. Víctor de la Peña and Evarist Giné. *Decoupling: from dependence to independence*. Springer, 1999.
11. Victor de la Peña and Stephen Montgomery-Smith. Decoupling inequalities for the tail probabilities of multivariate U-statistics. *Annals of Probability*, 23(2):806–816, 1995.
12. Irit Dinur. The PCP Theorem by gap amplification. *Journal of the ACM*, 54(3):1–44, 2007.

13 Irit Dinur, Ehud Friedgut, Guy Kindler, and Ryan O'Donnell. On the Fourier tails of bounded functions over the discrete cube. *Israel Journal of Mathematics*, 160(1):389–412, 2007.

14 David Ellis, Yuval Filmus, and Ehud Friedgut. Triangle-intersecting families of graphs. *Journal of the European Mathematical Society*, 14(3):841–885, 2012.

15 Ehud Friedgut. Boolean functions with low average sensitivity depend on few coordinates. *Combinatorica*, 18(1):27–36, 1998.

16 Ehud Friedgut and Gil Kalai. Every monotone graph property has a sharp threshold. *Proceedings of the American Mathematical Society*, 124(10):2993–3002, 1996.

17 Ehud Friedgut, Gil Kalai, and Assaf Naor. Boolean functions whose Fourier transform is concentrated on the first two levels and neutral social choice. *Advances in Applied Mathematics*, 29(3):427–437, 2002.

18 Evarist Giné. A consequence for random polynomials of a result of de la Peña and Montgomery-Smith. In *Probability in Banach Spaces 10*, volume 43 of *Progress in Probability*. Birkhäuser–Verlag, 1998.

19 Jacek Jendrej, Krzysztof Oleszkiewicz, and Jakub Wojtaszczyk. On some extensions of the FKN theorem. Manuscript, 2012. To appear in *Theory of Computation*.

20 Daniel Kane. k-independent Gaussians fool polynomial threshold functions. In *Proceedings of the 26th Annual Computational Complexity Conference*, pages 252–261, 2011.

21 Daniel Kane and Raghu Meka. A PRG for Lipschitz functions of polynomials with applications to Sparsest Cut. In *Proceedings of the 45th Annual ACM Symposium on Theory of Computing*, pages 1–10, 2013.

22 Subhash Khot. On the power of unique 2-prover 1-round games. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing*, pages 767–775, 2002.

23 Subhash Khot and Assaf Naor. Nonembeddability theorems via Fourier analysis. *Mathematische Annalen*, 334(4):821–852, 2006.

24 Subhash Khot and Assaf Naor. Linear equations modulo 2 and the L_1 diameter of convex bodies. *SIAM Journal on Computing*, 38(4):1448–1463, 2008.

25 Guy Kindler. *Property Testing, PCP, and juntas*. PhD thesis, Tel Aviv University, 2002.

26 Guy Kindler and Ryan O'Donnell. Gaussian noise sensitivity and Fourier tails. In *Proceedings of the 27th Annual Computational Complexity Conference*, pages 137–147, 2012.

27 Guy Kindler and Shmuel Safra. Noise-resistant Boolean functions are juntas. Manuscript, 2002.

28 Stanisław Kwapień. Decoupling inequalities for polynomial chaos. *Annals of Probability*, 15(3):1062–1071, 1987.

29 Shachar Lovett. An elementary proof of anti-concentration of polynomials in Gaussian variables. Technical Report 182, Electronic Colloquium on Computational Complexity, 2010.

30 Nathaniel Macon and Abraham Spitzbart. Inverses of Vandermonde matrices. *The American Mathematical Monthly*, 65:95–100, 1958.

31 Konstantin Makarychev and Maxim Sviridenko. Solving optimization problems with diseconomies of scale via decoupling. In *Proceedings of the 55th Annual IEEE Symposium on Foundations of Computer Science*, pages 571–580, 2014.

32 Elchanan Mossel, Ryan O'Donnell, and Krzysztof Oleszkiewicz. Noise stability of functions with low influences: invariance and optimality. *Annals of Mathematics*, 171(1):295–341, 2010.

33 Ryan O'Donnell. *Analysis of Boolean Functions*. Cambridge University Press, 2014.

Polynomials, Quantum Query Complexity, and Grothendieck's Inequality

Scott Aaronson[*1], **Andris Ambainis**[†2], **Jānis Iraids**[‡3], **Martins Kokainis**[§4], **and Juris Smotrovs**[¶5]

1 **Computer Science and Artificial Intelligence Laboratory, MIT, Cambridge, USA**
aaronson@csail.mit.edu
2 **Faculty of Computing, University of Latvia, Riga, Latvia**
ambainis@lu.lv
3 **Faculty of Computing, University of Latvia, Riga, Latvia**
janis.iraids@gmail.com
4 **Faculty of Computing, University of Latvia, Riga, Latvia**
martins.kokainis@lu.lv
5 **Faculty of Computing, University of Latvia, Riga, Latvia**
juris.smotrovs@lu.lv

Abstract

We show an equivalence between 1-query quantum algorithms and representations by degree-2 polynomials. Namely, a partial Boolean function f is computable by a 1-query quantum algorithm with error bounded by $\epsilon < 1/2$ iff f can be approximated by a degree-2 polynomial with error bounded by $\epsilon' < 1/2$. This result holds for two different notions of approximation by a polynomial: the standard definition of Nisan and Szegedy [21] and the approximation by block-multilinear polynomials recently introduced by Aaronson and Ambainis [1]. The proof uses Grothendieck's inequality to relate two matrix norms, with one norm corresponding to polynomial approximations and the other norm corresponding to quantum algorithms.

We also show two results for polynomials of higher degree. First, there is a total Boolean function which requires $\tilde{\Omega}(n)$ quantum queries but can be represented by a block-multilinear polynomial of degree $\tilde{O}(\sqrt{n})$. Thus, in the general case (for an arbitrary number of queries), block-multilinear polynomials are not equivalent to quantum algorithms.

Second, for any constant degree k, the two notions of approximation by a polynomial (the standard and the block-multilinear) are equivalent. As a consequence, we solve an open problem from [1], showing that one can estimate the value of any bounded degree-k polynomial $p : \{0,1\}^n \to [-1,1]$ with $O(n^{1-\frac{1}{2k}})$ queries.

1998 ACM Subject Classification F.2.3 Tradeoffs between Complexity Measures

Keywords and phrases quantum algorithms, Boolean functions, approximation by polynomials, Grothendieck's inequality

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.25

[*] Supported by an Alan T. Waterman Award from the National Science Foundation, under grant no. 1249349.
[†] Supported by the European Commission FET-Proactive project QALGO, ERC Advanced Grant MQC and Latvian State Research programme NexIT project No. 1.
[‡] Supported by the European Commission FET-Proactive project QALGO, ERC Advanced Grant MQC and Latvian State Research programme NexIT project No. 1.
[§] Supported by the European Commission FET-Proactive project QALGO, ERC Advanced Grant MQC and Latvian State Research programme NexIT project No. 1.
[¶] Supported by the European Commission FET-Proactive project QALGO, ERC Advanced Grant MQC and Latvian State Research programme NexIT project No. 1.

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 25; pp. 25:1–25:19

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

1 Introduction

Many of the known quantum algorithms can be studied in the query model where one measures the complexity of an algorithm by the number of queries to the input that it makes. In particular, this model encompasses Grover's search [17], the quantum part of Shor's factoring algorithm (period-finding) [25], their generalizations and many of the more recent quantum algorithms such as element distinctness [7] and NAND tree evaluation [16, 8, 24].

For proving lower bounds on quantum query algorithms, one often uses a connection to polynomials [9]. After k queries to an input $x_1, \ldots, x_N$, the amplitudes of the algorithm's quantum state are polynomials of degree at most k in $x_1, \ldots, x_N$. Therefore, one can prove that there is no quantum algorithm using fewer than k queries by showing the non-existence of a polynomial with certain properties.

For example, one can use this approach to show that any quantum algorithm for Grover's search algorithm requires $\Omega(\sqrt{N})$ queries [9] or to show an optimal quantum lower bound for finding collisions [4]. In some cases, the lower bounds obtained by polynomials method are tight, either exactly (for example, for computing the parity of N input bits $x_1, \ldots, x_N$ [9]) or up to a constant factor (Grover's search and many other examples). In other cases, the number of queries to compute a function $f(x_1, \ldots, x_N)$ is asymptotically larger than the lower bound which follows from polynomials [6, 3].

In this paper, we discover the first case where we can go in the opposite direction: from a polynomial to a bounded-error quantum algorithm[1]. That is, polynomials with certain properties and quantum algorithms are equivalent!

In more detail, we consider computing partial Boolean functions $f(x_1, \ldots, x_n)$ and show that the existence of a quantum algorithm that computes f with 1 query is equivalent to the existence of a degree 2 polynomial that approximates f. This result holds for two different notions of approximation by a polynomial: the standard one in [21] and the approximation by block-multilinear polynomials introduced in [1].

To transform a polynomial into a quantum algorithm, we first transform it into the block-multilinear form of [1] and then use a variant of Grothendieck's inequality for relating two matrix norms [23]. One of the two norms corresponds to the constraints on the block-multilinear polynomials while the other norm corresponds to algorithm's transformations being unitary. While Grothendieck's inequality has been used in the context of quantum non-locality (e.g. in [5]), this appears to be its first use in the context of quantum algorithms.

We then show two results for polynomials of larger degree:

- similarly to general polynomials, block-multilinear polynomials are not equivalent to quantum algorithms in the general case: one of *cheat-sheet* functions of [3] requires $\tilde{\Omega}(n)$ quantum queries but can be described by a block-multilinear polynomial of degree $\tilde{O}(\sqrt{n})$;
- for representations by polynomials of degree $d = O(1)$, a partial function f can be represented by a general polynomial of degree d if and only if it can be represented by a block-multilinear polynomial of degree d.

We note that the first result does not exclude an equivalence between quantum algorithms and polynomials for a small number of queries that is larger than 1. For example, 2-query quantum algorithms could be equivalent to polynomials of degree 4. The second result shows that, to prove such an equivalence, it suffices to give a transformation from block-multilinear polynomials to quantum algorithms.

[1] In unbounded-error settings, equivalences between quantum algorithms and polynomials were previously shown by de Wolf [26] and by Montanaro et al. [20].

Another consequence of the second result is that, if we have a general polynomial $f(x_1, \ldots, x_n)$ which is bounded (i.e., $|f| \leq 1$ for all $x_1, \ldots, x_n \in \{0, 1\}$), the value of this polynomial can be estimated with $O(n^{1-1/2d})$ queries about values of $x_1, \ldots, x_n$. This resolves an open problem from [1] and is shown by transforming f into a block-multilinear form and then using the sampling algorithm of [1] for block-multilinear polynomials. The second result and this consequence was discovered independently by us and O'Donnell and Zhao [15].

2 Preliminaries

2.1 Notation

By $[a\,..\,b]$, with a, b being integers, $a \leq b$, we denote the set $\{a, a+1, a+2, \ldots, b\}$. When $a = 1$, notation $[a\,..\,b]$ is simplified to $[b]$.

For a vector x, let $\|x\|_p$ stand for the p-norm; when $p = 2$, this is the Euclidean norm and the notation is simplified to $\|x\|$. For a matrix A, by $\|A\|_{p\to q}$ we denote

$$\|A\|_{p\to q} = \sup_{x:\|x\|_p\neq 0} \frac{\|Ax\|_q}{\|x\|_p} = \max_{x:\|x\|_p=1} \|Ax\|_q = \max_{x:\|x\|_p\leq 1} \|Ax\|_q .$$

By $\|A\|$ we understand the usual operator norm $\|A\|_{2\to 2}$.

D_x stands for the diagonal matrix with components of x on its diagonal.

By K we denote the (real) Grothendieck's constant which is defined as the smallest number with the following property: if $A = (a_{ij})$ is such that $\sum_{i,j} a_{ij} x_i y_j \leq 1$ for any choice of $x_i, y_j \in \{-1, 1\}$, then $\sum_{i,j} a_{ij} \langle u_i, v_j \rangle \leq K$ for any choice of vectors (with real components) u_i, v_j with $\|u_i\| = 1$ and $\|v_j\| = 1$ for all i, j. It is known [23, 11] that

$$\frac{\pi}{2} \leq K < \frac{\pi}{2\ln(1+\sqrt{2})}.$$

2.2 Quantum query complexity and polynomial degree

We consider computing partial Boolean functions $f(x_1, \ldots, x_n) : X \to \{0, 1\}$ (for some $X \subseteq \{0, 1\}^n$) in the standard quantum query model. For technical convenience, we relabel the values of input variables x_i from $\{0, 1\}$ to $\{-1, 1\}$. Then a partial Boolean function f maps a set $X \subseteq \{-1, 1\}^n$ to $\{0, 1\}$.

Let $\mathrm{Q}_\epsilon(f)$ be the minimum number of queries in a quantum algorithm computing f correctly with probability at least $1 - \epsilon$, for every $x = (x_1, \ldots, x_n)$ for which $f(x)$ is defined.

▶ **Definition 2.1.** $\widetilde{\deg}_\epsilon(f)$ is the minimum degree of a polynomial p (in variables $x_1, \ldots, x_n$) such that

1. $|p(x) - f(x)| \leq \epsilon$ for all $x \in \{-1, 1\}^n$ for which $f(x)$ is defined;
2. $p(x) \in [0, 1]$ for all $x \in \{-1, 1\}^n$.

$\deg(f)$ denotes $\widetilde{\deg}_0(f)$.

It is well known that $\mathrm{Q}_\epsilon(f) \geq \frac{1}{2}\widetilde{\deg}_\epsilon(f)$ [9]. We now consider a refinement of this result due to [1]. We say that a polynomial p of degree k is *block-multilinear* if its variables $x_1, \ldots, x_N$ can be partitioned into k blocks, $R_1, \ldots, R_k$, so that every monomial of p contains exactly one variable from each block[2]

[2] In other words, a block-multilinear polynomial is just a multilinear form. We, however, use the word

▶ **Lemma 2.2.** *[1, Lemma 20] Let $\mathcal{A}$ be a quantum algorithm that makes t queries to a Boolean input $x \in \{-1,1\}^n$. Then there exists a degree-$2t$ block-multilinear polynomial $p : \mathbb{R}^{2t(n+1)} \to \mathbb{R}$, with $2t$ blocks of $n+1$ variables each, such that*

(i) *the probability that $\mathcal{A}$ outputs 1 for an input $x = (x_1, \ldots, x_n) \in \{-1,1\}^n$ equals $p(\tilde{x}, \ldots, \tilde{x})$, where $\tilde{x} := (1, x_1, \ldots, x_n)$ (with $\tilde{x}$ repeated $2t$ times), and*

(ii) *$p(z) \in [-1,1]$ for all $z \in \{-1,1\}^{2t(n+1)}$.*

The first variable in each block (which is set to 1 in the requirement (i)) corresponds to the possibility that the algorithm is not asking any of the actual variables $x_1, \ldots, x_n$ in a given query. (Although the statement of Lemma 20 in [1] does not mention such variables explicitly, they are used in the proof of the Lemma.)

▶ **Definition 2.3.** Let the *block-multilinear approximate degree* of f, or $\widetilde{\mathrm{bmdeg}}_\epsilon(f)$, be the minimum degree of any block-multilinear polynomial $p : \mathbb{R}^{k(n+1)} \to \mathbb{R}$, with k blocks of $n+1$ variables each, such that

(i) $p(\tilde{x}, \ldots, \tilde{x}) \in [0,1]$ and $|p(\tilde{x}, \ldots, \tilde{x}) - f(x)| \leq \epsilon$ for all $x \in \{-1,1\}^n$ for which $f(x)$ is defined, and

(ii) $p(x_{1,0}, x_{1,1}, \ldots, x_{1,n}, x_{2,0}, \ldots, x_{k,n}) \in [-1,1]$ for all $x_{1,0}, \ldots, x_{k,n} \in \{-1,1\}^{k(n+1)}$.

$\mathrm{bmdeg}(f)$ denotes $\widetilde{\mathrm{bmdeg}}_0(f)$.

As a particular case, this definition includes block-multilinear polynomials $p : \mathbb{R}^{kn} \to \mathbb{R}$ which satisfy

$$\forall x \in \{-1,1\}^n \ \ |p(x, \ldots, x) - f(x)| \leq \epsilon \quad \text{and} \quad \forall z \in \{-1,1\}^{kn}\, p(z) \in [-1,1],$$

because we can view them as polynomials $p : \mathbb{R}^{k(n+1)} \to \mathbb{R}$ in which each monomial containing a variable $x_{1,0}$, $x_{2,0}$, …, or $x_{k,0}$ has a coefficient zero.

We have $\widetilde{\deg}_\epsilon(f) \leq \widetilde{\mathrm{bmdeg}}_\epsilon(f) \leq 2\,\mathrm{Q}_\epsilon(f)$. The first of the two inequalities follows by taking $q(x) = p(\tilde{x}, \ldots, \tilde{x})$. If p satisfies the requirements of Definition 2.3, then q satisfies the requirements of Definition 2.1. The second inequality follows from Lemma 2.2.

2.3 Equivalence between block-multlinear and general polynomials

The two types of polynomial representations ($\widetilde{\deg}$ and $\widetilde{\mathrm{bmdeg}}$) are equivalent to one another, up to some loss in the quality of approximation. This has been shown independently by us and by O'Donnell and Zhao [15]:

▶ **Theorem 2.4.** *Let $p(x_1, \ldots, x_n)$ be a polynomial of degree d. Then there is a block-multilinear polynomial $\tilde{p} : R^{(n+1)d} \to R$ such that*

1. *$\tilde{p}(\tilde{x}, \ldots, \tilde{x}) = p(x)$ for any $x \in \{-1,1\}^n$;*
2. *$|\tilde{p}(y)| \leq C_d$ for any $d \in \{-1,1\}^{(n+1)d}$ with C_d being a constant that depends on the degree d only.*

O'Donnell and Zhao [15] show $C_d \leq (2e)^d$. In the full version of this paper [2], we show our version of this result with $C_2 = 3$ for $d = 2$ and $C_d = O(3.5911...^d)$.

The result of O'Donnell and Zhao is a special case of the general theory of *decoupling* [18, 22] which proves much more general results. In contrast, our bounds are based on explicit

block-multilinear, to emphasize the difference from standard polynomial representations of Boolean functions which are multilinear but are not multilinear forms.

combinatorial arguments. These arguments are specific to the problem above but allow us to obtain better constants C_d.

As a consequence of this theorem, we have

▶ **Corollary 2.5.** *Let ϵ be such that $0 \leq \epsilon < \frac{1}{2}$ and let $\epsilon' = \frac{1}{2} - \frac{1}{C_d}(\frac{1}{2} - \epsilon)$. Then $\widetilde{\deg}_\epsilon(f) \leq d$ implies $\widetilde{\mathrm{bmdeg}}_{\epsilon'}(f) \leq d$.*

Proof. We take the polynomial q which approximates $f(x_1, \ldots, x_n)$ with error ϵ according to Definition 2.1 and apply Theorem 2.4 to $p(x_1, \ldots, x_n) = q(x_1, \ldots, x_n) - \frac{1}{2}$. Then the polynomial $\frac{1}{2} + \tilde{p}$ approximates f in the sense of Definition 2.3. ◀

2.4 Block-multilinear polynomials of degree 2

Let

$$p(x_1, \ldots, x_n, y_1, \ldots, y_m) = \sum_{\substack{i \in [n] \\ j \in [m]}} a_{ij} x_i y_j, \tag{1}$$

be a block-multilinear polynomial of degree 2, with the variables in the first block labeled as $x_1, \ldots, x_n$ and the variables in the second block labeled as $y_1, \ldots, y_m$. We say that p is *bounded* if $|p(x_1, \ldots, x_n, y_1, \ldots, y_m)| \leq 1$ for all $x_1, \ldots, y_m \in \{-1, 1\}$. Then we have

$$\max_{\substack{x \in \{-1,1\}^n \\ y \in \{-1,1\}^m}} \left| \sum_{\substack{i \in [n] \\ j \in [m]}} a_{ij} x_i y_j \right| \leq 1.$$

Let A be the $n \times m$ matrix with entries a_{ij}, then

$$p(x, y) = x^T A y \quad \text{for all } x \in \mathbb{R}^n,\ y \in \mathbb{R}^m$$

and p being bounded translates to the $\infty \to 1$ norm of A being at most 1, i.e., $\|A\|_{\infty \to 1} \leq 1$.

3 Equivalence between polynomials of degree 2 and 1-query quantum algorithms

Let f be a partial Boolean function. In this section, we show that the following two statements are equivalent[3]:

(a) $\mathrm{Q}_\epsilon(f) \leq 1$ for some ϵ with $0 \leq \epsilon < \frac{1}{2}$;

(b) $\widetilde{\mathrm{bmdeg}}_{\epsilon'}(f) \leq 2$ for some ϵ' with $0 \leq \epsilon' < \frac{1}{2}$;

Given (a), Lemma 2.2 implies that (b) holds with $\epsilon' = \epsilon$. We now show that (b) implies (a) with $\epsilon = \frac{K+\epsilon'}{2(K+1)}$ where K is Grothendieck's constant.

Because of results in Section 2.3, we also get a similar equivalence between $\mathrm{Q}_\epsilon(f) \leq 1$ and $\widetilde{\deg}_{\epsilon'}(f) \leq 2$.

▶ **Theorem 3.1.** *Let f be a partial Boolean function. If $\widetilde{\mathrm{bmdeg}}_{\epsilon'}(f) \leq 2$, then $\mathrm{Q}_\epsilon(f) \leq 1$ for $\epsilon = \frac{K+\epsilon'}{2(K+1)}$.*

[3] The equivalence here involves some loss in the error ϵ. However, the bound ϵ on the error probability of the resulting quantum algorithm only depends on the error of the polynomial approximation from which we started and does not increase with the number of variables n.

Proof. We start with two technical lemmas.

▶ **Lemma 3.2.** *If an $n \times m$ complex matrix B satisfies $\|B\| \leq C$, then there exists a unitary U (on a possibly larger space with basis states $|1\rangle, \ldots, |k\rangle$ for some $k \geq \max(n, m)$) such that, for any unit vector $|y\rangle = \sum_{i=1}^{m} \alpha_i |i\rangle$, $U|y\rangle = \frac{B|y\rangle}{C} + |\phi\rangle$, with $|\phi\rangle$ consisting of basis states $|i\rangle$, $i > n$ only.*

Proof. Without the loss of generality, we can assume that $C = 1$ (otherwise, we just replace the matrix B by $\frac{B}{C}$).

Let $A = I - B^\dagger B$. Since $\|B\| \leq 1$, the eigenvalues of $B^\dagger B$ are at most 1 and, hence, A is positive semidefinite. Let $A = V^\dagger \Lambda V$ be the eigendecomposition of A, with V being a unitary matrix and Λ a diagonal matrix. We take $W = \sqrt{\Lambda} V$. Then $A = W^\dagger W$ and, if we take the block matrix $U = \begin{pmatrix} B \\ W \end{pmatrix}$, we get $U^\dagger U = B^\dagger B + W^\dagger W = I$.

Let $k \times m$ be the size of the matrix U. For any $i \in \{1, \ldots, m\}$, we have $\langle i|U^\dagger U|i\rangle = \langle i|I|i\rangle = 1$ and for any $i, j \in \{1, \ldots, m\} : i \neq j$, we have $\langle i|U^\dagger U|j\rangle = \langle i|I|j\rangle = 0$. Therefore, $U|1\rangle, \ldots, U|m\rangle$ are orthogonal vectors of length 1 and we can complete U to a $k \times k$ unitary matrix by choosing $U|m+1\rangle, \ldots, U|k\rangle$ so that they are orthogonal (both one to another and to $U|1\rangle, \ldots, U|m\rangle$) and of length 1. ◀

▶ **Lemma 3.3.** *Let $A = (a_{ij})_{i \in [n], j \in [m]}$ be a real matrix with $\sqrt{nm}\|A\| \leq C$ and let*

$$p(x_1, \ldots, x_n, y_1, \ldots, y_m) = \sum_{i=1}^{n} \sum_{j=1}^{m} a_{ij} x_i y_j.$$

Then there is a quantum algorithm that makes 1 query to $x_1, \ldots, x_n$, $y_1, \ldots, y_m$ and outputs 1 with probability

$$r = \frac{1}{2}\left(1 + \frac{p(x_1, \ldots, x_n, y_1, \ldots, y_m)}{C}\right).$$

Proof. Let $B = \sqrt{nm}\, A$, $A = (a_{ij})$. Then

$$\|B\| = \|A\| \sqrt{nm} \leq C.$$

The 1-query quantum algorithm uses a version of the well-known SWAP test [12] for estimating the inner product $|\langle \psi | \psi' \rangle|$ of two quantum states $|\psi\rangle$ and $|\psi'\rangle$. Our test works by preparing the state

$$\frac{1}{\sqrt{2}}|0\rangle|\psi\rangle + \frac{1}{\sqrt{2}}|1\rangle|\psi'\rangle \tag{2}$$

and then performing the Hadamard transformation on the first qubit and measuring the first qubit[4]. The probability that the result of the measurement is 0 is equal to

$$r = \frac{1}{2}\left(1 + Re\left(\langle \psi | \psi' \rangle\right)\right)$$

where $Re(x)$ denotes the real part of a complex number x.

[4] This test is slightly different from the standard SWAP test in which one prepares both $|\psi\rangle$ and $|\psi'\rangle$ and then performs a SWAP gate conditioned by a qubit that is initially in the $\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$ state. Because of this difference, we can perform the SWAP test with just 1 query instead of 2 (one for $|\psi\rangle$ and one for $|\psi'\rangle$). Another result of this difference is that the probability of measuring 0 changes from $\frac{1}{2}(1 + |\langle \psi | \psi' \rangle|^2)$ for the standard SWAP test to $\frac{1}{2}\left(1 + Re(\langle \psi | \psi' \rangle)\right)$ for our test.

By Lemma 3.2, there is a unitary U s.t. for any unit vector $|y\rangle = \sum_{i=1}^{m} \alpha_i |i\rangle$ we have $U|y\rangle = \frac{B|y\rangle}{C} + |\phi\rangle$, with $\langle i|\phi\rangle = 0$ for all $i \in [n]$.

The algorithm applies SWAP test to $|x\rangle = \frac{1}{\sqrt{n}} \sum_{i=1}^{n} x_i |i\rangle$ and $U|y\rangle$, $|y\rangle = \frac{1}{\sqrt{m}} \sum_{i=1}^{m} y_i |i\rangle$. Each of those states can be prepared with one query (to x_i's or y_i's). Hence, we can also prepare the state (2) with one query. The inner product $\langle\psi|\psi'\rangle$ that is being estimated is equal to

$$\langle x|U|y\rangle = \frac{1}{C}\langle x|B|y\rangle = \frac{1}{C}p(x_1, \ldots, x_n, y_1, \ldots, y_m). \quad \blacktriangleleft$$

Let $p(x_1, \ldots, x_n, y_1, \ldots, y_m) = \sum_{i=1}^{n}\sum_{j=1}^{m} a_{ij}x_iy_j$ be the polynomial from Definition 2.3 which shows that $\widetilde{\text{bmdeg}}_{\epsilon'}(f) = 2$. Then as we argued in Section 2.4, the matrix $A = (a_{ij})$ satisfies $\|A\|_{\infty\to 1} \leq 1$. Although this does not imply that $\|A\|$ is sufficiently small, we can preprocess the polynomial p so that we achieve $\sqrt{n'm'}\|A'\| \leq K$ for the n'-by-m' matrix A' of coefficients of the polynomial after the preprocessing.

To preprocess the polynomial, we perform an operation called *variable-splitting* [1]. The operation consists of taking a variable x_j (or y_j) and replacing it by m variables, in the following way. We introduce m new variables $x_{l_1}, \ldots, x_{l_m}$, and define p' as the polynomial obtained by substituting $\frac{x_{l_1}+\cdots+x_{l_m}}{m}$ in the polynomial p instead of x_j. If we substitute $x_{l_1} = \ldots = x_{l_m} = x_j$, p' is equal to $p(x_1, \ldots, x_n, y_1, \ldots, y_m)$. Thus, being able to evaluate p' implies being able to evaluate p (in the same sense of the word "evaluate").

In Appendix A, we show

▶ **Lemma 3.4.** *If a polynomial*

$$p(x_1, \ldots, x_n, y_1, \ldots, y_m) = \sum_{i=1}^{n}\sum_{j=1}^{m} a_{ij}x_iy_j$$

satisfies $p(x, y) \in [-1, 1]$ for all $x \in \{-1, 1\}^n$, $y \in \{-1, 1\}^m$, then for every $\delta > 0$ there exists a sequence of row and column splittings that transforms $A = (a_{ij})$ to an $n' \times m'$ matrix $A' = (a'_{ij})$ that satisfies

$$\frac{\|A'\|\sqrt{n'm'}}{\|A'\|_{\infty\to 1}} \leq K + \delta.$$

Then we can apply Lemma 3.3 with $C = K + \delta$ to evaluate the polynomial

$$p'(x'_1, \ldots, x'_{n'}, y'_1, \ldots, y'_{m'}) = \sum_{i=1}^{n'}\sum_{j=1}^{m'} a_{ij}x'_iy'_j.$$

for $(x'_1, \ldots, x'_{n'}, y'_1, \ldots, y'_{m'})$ which corresponds to the point $(x_1, \ldots, x_n, y_1, \ldots, y_m)$ at which we want to evaluate the original polynomial $p(x_1, \ldots, y_m)$.

If $p(x, y) \in [0, \epsilon']$, then Lemma 3.3 gives $r \leq (1 + \frac{\epsilon'}{K})/2$. If $p(x, y) \in [1 - \epsilon', 1]$, then $r \geq (1 + \frac{1-\epsilon'}{K})/2$.

We now consider an algorithm which outputs 0 with probability $\frac{1}{2K+1}$ and runs the algorithm of Lemma 3.3 otherwise (with probability $\frac{2K}{2K+1}$). Let q be the probability of this algorithm outputting 1. If $p(x, y) \in [0, \epsilon']$, then $q = \frac{2K}{2K+1}r \leq \frac{K+\epsilon'}{2K+1}$. If $p(x, y) \in [1 - \epsilon', 1]$, then $q = \frac{2K}{2K+1}r \geq \frac{K+1-\epsilon'}{2K+1}$. Thus, we have a quantum algorithm with a probability of error which is at most $\epsilon = \frac{K+\epsilon'}{2K+1}$. ◀

4 Results on polynomials of higher degrees

4.1 bmdeg and deg vs. Q

The biggest known separation between deg and Q is $Q_\epsilon(f) = \tilde{\Omega}(\deg^2(f))$, recently shown by Aaronson et al. [3] using a novel *cheat-sheet* technique. We extend this result to

▶ **Theorem 4.1.** *There exists f with $Q_\epsilon(f) = \tilde{\Omega}(\mathrm{bmdeg}^2(f))$.*

Proof. In Appendix B. ◀

Aaronson et al. [3] also show a separation $Q_\epsilon(f) = \tilde{\Omega}(\widetilde{\deg}(f))^4)$ which does not seem to give $Q_\epsilon(f) = \tilde{\Omega}(\widetilde{\mathrm{bmdeg}}(f))^4)$. (For the natural way of transforming the approximating polynomial of [3] into a block-multilinear form, the resulting block-multilinear polynomial $p(z^{(1)}, z^{(2)}, \ldots)$ can take values that are exponentially large (in its degree) if the blocks $z^{(1)}, z^{(2)}, \ldots$ are not all equal.)

Because of Theorem 4.1, there is no transformation from a polynomial of degree $2k$ that approximates $f(x_1, \ldots, x_n)$ with error $\epsilon < 1/2$ to a quantum algorithm with k queries and error $\epsilon' < 1/2$, with ϵ and ϵ' independent of k.

However, there may be a transformation from polynomials of degree $2k$ to quantum algorithms with k queries, with the error $\epsilon' = g(\epsilon, k)$ of the resulting quantum algorithm depending on k but not on function $f(x_1, \ldots, x_n)$ or the number of variables n.

Theorem 4.1 implies the following limit on such transformations:

▶ **Theorem 4.2.** *There is a sequence of Boolean functions $f_1, f_2, \ldots$ such that, for any sequence of quantum algorithms $\mathcal{A}_1, \mathcal{A}_2, \ldots$ computing them with $O(\mathrm{bmdeg}(f_i))$ queries, the probability of correct answer is at most*

$$\frac{1}{2} + O\left(\frac{1}{\mathrm{bmdeg}(f_i)}\right).$$

Proof. Let f be the function from Theorem 4.1. Then we have $\mathrm{bmdeg}(f) = \tilde{O}(\sqrt{n})$.

If we have a quantum algorithm $\mathcal{A}$ that computes a function f with a probability of correct answer at least $\frac{1}{2} + \delta$, we can use amplitude estimation [10] to estimate whether $\mathcal{A}$ produces answer $f = 1$ with probability at least $\frac{1}{2} + \delta$ or with probability at most $\frac{1}{2} - \delta$. The standard analysis of amplitude estimation [10] shows that we can obtain an estimate that is correct with probability at least $2/3$, with $O(1/\delta)$ repetitions of $\mathcal{A}$. To avoid a contradiction with $Q_\epsilon(f) = \Omega(n)$, we must have

$$\frac{\sqrt{n}}{\delta} = \Omega(n)$$

which implies $\delta = O(\frac{1}{\sqrt{n}})$. ◀

A result with a weaker bound on the error is, however, possible. For example, it is possible that $\widetilde{\deg}_{1/2-\delta}(f) = 2k$ or $\widetilde{\mathrm{bmdeg}}_{1/2-\delta}(f) = 2k$ implies a quantum algorithm which makes k queries and has the error probability at most $\frac{1}{2} - \Omega(\frac{\delta}{2^k})$ or at most $\frac{1}{2} - \Omega(\frac{\delta}{k^2})$.

4.2 Equivalence between general and block-multilinear polynomials

By Corollary 2.5, $\widetilde{\deg}_\epsilon(f) \leq d$ implies $\widetilde{\mathrm{bmdeg}}_{\epsilon'}(f) \leq d$ with ϵ' that depends on ϵ and d only. Therefore, if we want to extend the equivalence between quantum algorithms and polynomials

to larger $d = O(1)$, it suffices to show how to transform block-multilinear polynomials into quantum algorithms.

Also, Aaronson and Ambainis [1] showed that a quantum algorithm which makes d queries can be simulated by a classical algorithm making $O(n^{1-1/2d})$ queries, based on the following result:

▶ **Theorem 4.3** ([1]). *Let $h : \mathbb{R}^{d(n+1)} \to \mathbb{R}$ be a block-multilinear polynomial of degree d with $|h(y)| \leq 1$ for any $y \in \{-1,1\}^{d(n+1)}$. Then $h(y)$ can be approximated within precision $\pm\epsilon$ with high probability, by querying $O((\frac{n}{\epsilon^2})^{1-1/d}))$ variables (with a big-O constant that is allowed to depend on d).*

It has been open whether a similar theorem holds for general (not block-multilinear) polynomials $h(x_1, \ldots, x_n)$. Aaronson and Ambainis [1] showed that this is true for degree 2 (using quite sophisticated tools from Fourier analysis) but left it as an open problem for higher degrees. With Theorem 2.4, we can immediately resolve this problem.

▶ **Corollary 4.4.** *Let $g : \mathbb{R}^n \to \mathbb{R}$ be a polynomial of degree d with $|g(y)| \leq 1$ for any $y \in \{-1,1\}^{d(n+1)}$. Then $g(y)$ can be approximated within precision $\pm\epsilon$ with high probability, by querying $O((\frac{n}{\epsilon^2})^{1-1/d}))$ variables (with a big-O constant that is allowed to depend on d).*

Proof. We apply Theorem 2.4 to construct a corresponding block-multilinear polynomial h and then use Theorem 4.3 to estimate h with precision $\frac{\epsilon}{B(d)}$. Since $B(d)$ is a constant for any fixed d, we can absorb it into the big-O constant. ◀

This result was independently shown by O'Donnell and Zhao [15] (using their form of Theorem 2.4) and us (using our version of Theorem 2.4, described in the full version of our paper [2]).

5 Conclusions

We have shown a new equivalence between quantum algorithms and polynomials: the existence of a 1-query quantum algorithm computing a partial Boolean function f is equivalent to the existence of a degree-2 polynomial p that approximates f. Our equivalence theorem can be seen as a counterpart of the equivalence between unbounded-error quantum algorithms and threshold polynomials, proved by Montanaro et al. [20], and the equivalence between nondeterministic quantum algorithms and nondeterministic polynomials, proved by de Wolf [26].

Our equivalence is, however, much more challenging to prove. A transformation from polynomials to unbounded-error or nondeterministic quantum algorithms can incur a very large loss in error probability (for example, it can transform a polynomial p with error $1/3$ to a quantum algorithm $\mathcal{A}$ with the probability of correct answer $\frac{1}{2} + \frac{1}{2^n}$). In contrast, our transformation produces a quantum algorithm whose error probability only depends on the approximation error of the polynomial p and not on the number of variables n. To achieve this, we use a relation between two matrix norms related to Groethendieck's inequality.

Our equivalence holds for two notions of approximability by a polynomial: the standard one [21] which allows arbitrary polynomials of degree 2 and the approximation by block-multilinear polynomials recently introduced by [1]. The first notion of approximability is known not to be equivalent to the existence of a quantum algorithm: there are several constructions of f for which $Q_\epsilon(f)$ is asymptotically larger than $\deg(f)$ [6, 3], with $Q_\epsilon(f) = \tilde{\Omega}(\deg^2(f))$ as the biggest currently known gap [3]. We have shown that a similar gap holds for the second

notion of approximability. Thus, neither of the two notions is equivalent to the existence of a quantum algorithm when the degree of a polynomial becomes large.

Three open problems are:

1. **Equivalence between quantum algorithms and polynomials for more than 1 query?** Is it true that quantum algorithms with 2 queries are equivalent to polynomials of degree 4? It is even possible that quantum algorithms with k queries are equivalent to polynomials of degree $2k$ for any constant k - as long as the relation between the error of quantum algorithm and the error of the polynomial approximation depends on k, as discussed in Section 4.1.
2. **From polynomials to quantum algorithms.** It would also be interesting to have more results about transforming polynomials into quantum algorithms, even if such results fell short of a full equivalence between the two notions. For example, if it were possible to transform polynomials of degree 3 into 2-query quantum algorithms this would be an interesting result, even though it would be short of being an equivalence (since 2 query quantum algorithms are transformable into polynomials of degree 4 and not 3).
3. **Other notions of approximability by polynomials?** Until this work, there was a hope that the block-multilinear polynomial degree $\widetilde{\text{bmdeg}}(f)$ may provide a tight characterization of the quantum query complexity $Q_\epsilon(f)$. Now, we know that the gap between $\text{bmdeg}(f)$ and $Q_\epsilon(f)$ can be as large as the best known gap between $\deg(f)$ and $Q_\epsilon(f)$. Can one come up with a different notion of polynomial degree that would be closer to $Q_\epsilon(f)$ than $\deg(f)$ or $\text{bmdeg}(f)$?

References

1 S. Aaronson, A. Ambainis. Forrelation: A Problem that Optimally Separates Quantum from Classical Computing. *Proceedings of STOC'15*, pp. 307–316. Also arxiv:1411.5729.

2 S. Aaronson, A. Ambainis, J. Iraids, M. Kokainis, J. Smotrovs. Polynomials, Quantum Query Complexity, and Grothendieck's Inequality. arxiv:1511.08682.

3 S. Aaronson, S. Ben-David, R. Kothari. Separations in query complexity using cheat sheets. *Proceedings of STOC'16*, to appear. arXiv:1511.01937.

4 S. Aaronson, Y. Shi. Quantum lower bounds for the collision and the element distinctness problems. *Journal of the ACM*, 51(4):595–605, 2004.

5 A. Acin, N Gisin, B Toner. Grothendieck's constant and local models for noisy entangled quantum states *Physical Review A*, 73 (6), 062105, 2006. Also quant-ph/0606138.

6 A. Ambainis. Polynomial degree vs. quantum query complexity. *Journal of Computer and System Sciences*, 72(2):220–238, 2006. Earlier versions at FOCS'03 and quant-ph/0305028.

7 A. Ambainis. Quantum walk algorithm for element distinctness. *SIAM Journal on Computing*, 37(1):210–239, 2007. Also FOCS'04 and quant-ph/0311001.

8 A. Ambainis, A. Childs, B. Reichardt, R. Spalek, S. Zhang. Any AND-OR Formula of Size N Can Be Evaluated in Time $N^{1/2+o(1)}$ on a Quantum Computer. *SIAM Journal on Computing*, 39(6):2513–2530, 2010. Also FOCS'07.

9 R. Beals, H. Buhrman, R. Cleve, M. Mosca, and R. de Wolf. Quantum lower bounds by polynomials. *Journal of the ACM*, 48(4):778–797, 2001. Earlier versions at FOCS'98 and quant-ph/9802049.

10 G. Brassard, P. Høyer, M. Mosca, A. Tapp. Quantum amplitude amplification and estimation. In *Quantum Computation and Quantum Information Science*, AMS Contemporary Mathematics Series, 305:53–74, 2002. Also quant-ph/0005055.

11 M. Braverman, K. Makarychev, Y. Makarychev, A. Naor. The Grothendieck Constant is Strictly Smaller than Krivine's Bound. *Proceedings of FOCS'2011*, pp. 453–462.

12 H. Buhrman, R. Cleve, J. Watrous, R. de Wolf. Quantum fingerprinting. *Physical Review Letters*, 87(16):167902, 2001. Also quant-ph/0102001.

13 H. Buhrman, R. de Wolf. Complexity measures and decision tree complexity: a survey. *Theoretical Computer Science*, 288:21–43, 2002.

14 J. Clauser, M. Horne, A. Shimony, R. Holt. Proposed experiment to test local hidden-variable theories. *Physical Review Letters*, 23(15):880, 1969.

15 R. O'Donnell and Y. Zhao. Polynomial bounds for decoupling, with applications. *Proceedings of CCC'2016.* Also CoRR abs/1512.01603.

16 E. Farhi, J. Goldstone, S. Gutmann, A Quantum Algorithm for the Hamiltonian NAND Tree. *Theory of Computing*, 4:169–190, 2008. Also quant-ph/0702144.

17 L. K. Grover. A fast quantum mechanical algorithm for database search. *Proceedings of STOC'96*, pp. 212–219. Also quant-ph/9605043.

18 S. Kwapien. Decoupling inequalities for polynomial chaos. *The Annals of Probability*, 15:1062–1071, 1987.

19 N. Linial, A. Shraibman. Lower bounds in communication complexity based on factorization norms. *Random Structures and Algorithms*, 34(3):368–394, 2009.

20 A. Montanaro, H. Nishimura, R. Raymond. Unbounded error quantum query complexity. *Theoretical Computer Science*, 412(35):4619–4628, 2011. Also arXiv:0712.1446.

21 N. Nisan, M. Szegedy. On the Degree of Boolean Functions as Real Polynomials. *Computational Complexity*, 4:301–313, 1994.

22 V. de la Pena, E. Gine. *Decoupling: from Dependence to Independence.* Springer, 1999.

23 G. Pisier. Grothendieck's theorem, past and present. *Bull. Am. Math. Soc., New Ser.*, 49(2):237–323, 2012.

24 B. Reichardt. Span-program-based quantum algorithm for evaluating unbalanced formulas. *Proceedings of TQC'11*, pp. 73–103. Also arXiv:0907.1622.

25 P. Shor. Algorithms for Quantum Computation: Discrete Logarithms and Factoring. *SIAM Journal on Computing*, 26:1484–1509, 1997. Also FOCS'94 and quant-ph/9508027.

26 R. de Wolf. Nondeterministic quantum query and quantum communication complexities. *SIAM Journal on Computing*, 32(3):681–699, 2003. Also arXiv:cs/0001014.

A Proof of Lemma 3.4

A.1 Additional Notation

The variables of the polynomial (1) correspond to rows and columns of the coefficient matrix $A = (a_{ij})$, $i \in [n]$, $j \in [m]$. Hence, we can describe variable-splitting in terms of rows and columns of A, introducing the operations of *row-splitting* and *column-splitting*.

Let $a_{i\cdot}$ stand for the ith row $(a_{i1}, \ldots, a_{im})$ of A and similarly $a_{\cdot j}$ stand for the jth column of A. Row-splitting (into k rows) takes a row $a_{i\cdot}$ and replaces it with k equal rows $a_{i\cdot}/k = (a_{i1}/k, \ldots, a_{im}/k)$. Similarly, column-splitting takes a column $a_{\cdot j}$ and replaces it with k equal columns $a_{\cdot j}/k$.

We also denote

$$\|A\|_G = \sup_{r\in\mathbb{N}} \sup_{\substack{p_i,q_j\in\mathbb{R}^r \\ \forall i:\|p_i\|=1 \\ \forall j:\|q_j\|=1}} \sum_{i,j} a_{ij} \langle p_i, q_j\rangle. \tag{3}$$

$\|\cdot\|_G$ is the dual norm of the factorization norm γ_2, see, e.g., [19].

Let $\lambda_{\max}(B)$ denote the maximal eigenvalue of a square matrix B; then

$$\|A\|^2 = \lambda_{\max}\left(A^T A\right) = \lambda_{\max}\left(AA^T\right). \tag{4}$$

Denote $g(A) = \sqrt{nm}\,\|A\| / \|A\|_{\infty\to 1}$. By $\Gamma(A)$ we denote the numerator $\|A\|\sqrt{nm}$.

We say that a matrix A' of size $n' \times m'$ can be obtained from A if there exists a sequence of row and column splittings that transforms A to the matrix A'; we denote it by $A \longrightarrow A'$. Moreover, for simplicity we assume that no row or column is split repeatedly, i.e., if a row $a_{i\cdot}$ is split into k rows $a_{i\cdot}/k$, then none of these obtained rows is split again.

Denote by $G(A)$ the infimum of $g(A')$ over all matrices A' which can be obtained from A:

$$G(A) := \inf_{A' : A \longrightarrow A'} g(A').$$

We have $g(A) \geq 1$ for all matrices A. (To see this, observe that $\frac{\|Ax\|_1}{\|x\|_\infty} \leq \frac{\sqrt{n}\|Ax\|_2}{\|x\|_2/\sqrt{m}} = \sqrt{nm}\frac{\|Ax\|_2}{\|x\|_2}$. Taking maximums over all x on both sides gives $\|A\|_{\infty\to 1} \leq \sqrt{nm}\|A\|$ which is equivalent to $g(A) \geq 1$.) Therefore, we also have $G(A) \geq 1$.

The assumption that no row or column is split repeatedly does not alter the value of this infimum; more generally, one could consider weighted splitting of rows (or columns), e.g., allowing to replace a row $a_{i\cdot}$ with k rows $w_j a_{i\cdot}$, $j \in [k]$, where w_j are non-negative weights satisfying $w_1 + \ldots + w_k = 1$. It is possible to show also in this case that the infimum of $g(A')$ over all matrices A', yielded by permitted splittings, has the same value as $G(A)$.

Let $\mathcal{A}$ denote the class of all matrices (with real entries) which do not contain zero rows or columns. Notice that if $A \in \mathcal{A}$ and $A \longrightarrow A'$, then also $A' \in \mathcal{A}$. The class $\mathcal{A}_{n,m}$ contains all matrices in $\mathcal{A}$ of size $n \times m$.

By $\mathbb{R}^n_+$ we denote the set of all vectors $w \in \mathbb{R}^n$ such that $w_i > 0$ for all $i \in [n]$.

Using the introduced notation, we can restate Lemma 3.4:

▶ **Lemma A.1.** *For every matrix A we have*

$$G(A) = \frac{\|A\|_G}{\|A\|_{\infty\to 1}} \leq K. \tag{5}$$

The inequality here is due to Grothendieck's inequality, see, e.g., Theorem 4 of [19]. The remaining part of this section is devoted to proving the equality in (5).

A.2 Splitting preserves infinity-to-one and Grothendieck's norms

Here we show that splitting rows or columns does not change the norms $\|\cdot\|_{\infty\to 1}$ and $\|\cdot\|_G$.

▶ **Lemma A.2.** *For every matrix $A \in \mathcal{A}$ and every A' s.t. $A \longrightarrow A'$ we have*

$$\|A\|_{\infty\to 1} = \|A'\|_{\infty\to 1} \quad \textit{and} \quad \|A\|_G = \|A'\|_G\,.$$

Proof. Let a matrix $A \in \mathcal{A}_{n,m}$ be fixed. It is sufficient to show the statement for matrices A' that can be obtained by splitting a row $a_{i\cdot}$ of A into $l+1$ rows $a_{i\cdot}/(l+1)$ (these rows are indexed by $i, \ldots, i+l$ in A'). We have

$$\|A\|_{\infty\to 1} = \max_{x:\|x\|_\infty \leq 1} \|Ax\|_1 = \max_{x\in\{-1,1\}^n} \|Ax\|_1 = \max_{\substack{x\in\{-1,1\}^n \\ y\in\{-1,1\}^m}} x^T A y.$$

Suppose that $x \in \{-1,1\}^n, y \in \{-1,1\}^m$ are such that $x^T A y = \|A\|_{\infty\to 1}$. Notice that

$$x^T A y = \sum_{k=1}^{n} x_k a_{k\cdot} y.$$

Let $x' \in \{-1,1\}^{n+l}$ be obtained from x by replacing x_i with $(x_i, x_i, \ldots, x_i)$ (i.e., the component x_i, corresponding to the split row $a_{i\cdot}$, is replicated $l+1$ times) and these components are indexed with $i, \ldots, i+l$ in x'. Then

$$(x')^T A' y = \sum_{k=1}^{n+l} x'_k a'_{k\cdot} y = (l+1) \cdot x_i \frac{a_{i\cdot}}{l+1} y + \sum_{k \neq i} x_k a_{k\cdot} y = \sum_{k=1}^{n} x_k a_{k\cdot} y = \|A\|_{\infty\to 1}.$$

This shows that $\|A'\|_{\infty\to 1} \geq \|A\|_{\infty\to 1}$.

Suppose that $x \in \{-1,1\}^{n+l}, y \in \{-1,1\}^m$ are such that $x^T A' y = \|A'\|_{\infty\to 1}$ and the rows $a'_{i'\cdot}$, $i' \in [i\,..\,i+l]$, are the rows $a_{i\cdot}/(l+1)$, obtained from $a_{i\cdot}$. Let $\tilde{x} \in \mathbb{R}^n$ be such that

$$\tilde{x}_k = \begin{cases} x_k & k = 1, 2, \ldots, i-1, \\ x_{k+l} & k = i+1, i+2, \ldots, n, \\ \frac{x_i + \ldots + x_{i+l}}{l+1}, & k = i. \end{cases}$$

Notice that $|\tilde{x}_i| \leq \frac{1}{l+1} \sum_{k=i}^{i+l} |x_k| = 1$. Thus $\|\tilde{x}\|_\infty \leq 1$. On the other hand,

$$\tilde{x}^T A y = \sum_{k=1}^{n} \tilde{x}_k a_{k\cdot} y = \frac{\sum\limits_{k \in [i\,..\,i+l]} x_k}{l+1} a_{i\cdot} y + \sum_{k=1}^{i-1} x_k a_{k\cdot} y + \sum_{k=i+1}^{n} x_{k+l} a_{k\cdot} y = \sum_{k=1}^{n+l} x_k a'_{k\cdot} y = \|A'\|_{\infty\to 1}.$$

Since

$$\|A\|_{\infty\to 1} = \sup_{\substack{x \in \mathbb{R}^n, y \in \mathbb{R}^m, \\ \|x\|_\infty \leq 1, \\ \|y\|_\infty \leq 1}} x^T A y,$$

this implies that $\|A\|_{\infty\to 1} \geq \|A'\|_{\infty\to 1}$. Hence the two norms are equal.

We can argue similarly with the norm $\|A\|_G$, see (3). Let unit vectors p_k, q_j (in $\mathbb{R}^r$ for some $r \in \mathbb{N}$) be fixed, $k \in [n]$, $j \in [m]$. Choose $n+l$ unit vectors as follows:

$$p'_k = \begin{cases} p_k, & k < i, \\ p_{k-l}, & k = i+l+1, \ldots, n+l, \\ p_i, & k \in [i\,..\,i+l]. \end{cases}$$

Then

$$\|A'\|_G \geq \sum_{k,j} a'_{kj} \langle p'_k, q_j \rangle = \sum_{k,j} a_{kj} \langle p_k, q_j \rangle.$$

Taking supremum over all r and unit vectors p_k, q_j, we obtain $\|A'\|_G \geq \|A\|_G$.

Now, let unit vectors p_k, q_j (in $\mathbb{R}^r$ for some $r \in \mathbb{N}$) be fixed, $k \in [n+l]$, $j \in [m]$.

Choose n vectors $\tilde{p}_k$ as follows:

$$\tilde{p}_k = \begin{cases} p_k, & k < i, \\ p_{k+l}, & k = i+1, \ldots, n, \\ \frac{p_i + \ldots + p_{i+l}}{l+1}, & k = i. \end{cases}$$

By the triangle inequality $\|\tilde{p}_i\| \leq \frac{1}{l+1} \sum_{k=i}^{i+l} \|p_k\| = 1$. Since

$$\|A\|_G = \sup_{r \in \mathbb{N}} \sup_{\substack{p_k, q_j \in \mathbb{R}^r \\ \forall k: \|p_k\| = 1 \\ \forall j: \|q_j\| = 1}} \sum_{k,j} a_{kj} \langle p_k, q_j \rangle = \sup_{r \in \mathbb{N}} \sup_{\substack{p_k, q_j \in \mathbb{R}^r \\ \forall k: \|p_k\| \leq 1 \\ \forall j: \|q_j\| \leq 1}} \sum_{k,j} a_{kj} \langle p_k, q_j \rangle,$$

we have $\sum_k \sum_j a_{kj} \langle \tilde{p}_k, q_j \rangle \leq \|A\|_G$.

It follows that

$$\sum_{k,j} a'_{kj} \langle p_k, q_j \rangle = \sum_{k \notin [i \,..\, i+l]} \sum_j a'_{kj} \langle p_k, q_j \rangle + \frac{1}{l+1} \sum_{k=i}^{i+l} \sum_j a_{ij} \langle p_k, q_j \rangle$$
$$= \sum_k \sum_j a_{kj} \langle \tilde{p}_k, q_j \rangle \leq \|A\|_G .$$

Taking the supremum over all r and p_k, q_j, we obtain $\|A\|_G \geq \|A'\|_G$.

Hence the two norms are equal. ◀

A.3 Characterization of row(column)-splitting

▶ **Lemma A.3.** *Suppose that $A \in \mathcal{A}_{n,m}$; for each $i \in [n]$ the row $a_{i\cdot}$ is split into k_i rows and for each $j \in [m]$ the column $a_{\cdot j}$ is split into l_j rows; the resulting matrix is denoted by A'.*

Then $\Gamma(A') = \left\|\tilde{A}\right\| \|w\| \|v\|$, where $\tilde{A} = (\tilde{a}_{ij})$,

$$\tilde{a}_{ij} = \frac{a_{ij}}{w_i v_j}, \ i \in [n], \ j \in [m],$$
$$w_i = \sqrt{k_i}, \quad v_j = \sqrt{l_j}.$$

Proof. The matrix A' is of size $(k_1 + \ldots + k_n) \times (l_1 + \ldots + l_m) = \|w\|^2 \|v\|^2$. Hence it is sufficient to show that $\|A'\| = \left\|\tilde{A}\right\|$.

We begin by showing this statement in case when $l_1 = l_2 = \ldots = l_m = 1$, i.e., only row-splitting takes place.

Denote $M_i = a_{i\cdot}^T a_{i\cdot}$. By (4),

$$\left\|\tilde{A}\right\|^2 = \lambda_{\max}(\tilde{A}^T \tilde{A}), \qquad \|A'\|^2 = \lambda_{\max}(A'^T A'),$$

Notice that

$$\tilde{A}^T \tilde{A} = \begin{pmatrix} w_1^{-1} a_{1\cdot}^T & w_2^{-1} a_{2\cdot}^T & \ldots & w_n^{-1} a_{n\cdot}^T \end{pmatrix} \begin{pmatrix} w_1^{-1} a_{1\cdot} \\ w_2^{-1} a_{2\cdot} \\ \ldots \\ w_n^{-1} a_{n\cdot} \end{pmatrix} = \sum_{i=1}^n w_i^{-2} M_i.$$

Similarly it can be obtained that

$$A'^T A' = \sum_{i=1}^n \sum_{j=1}^{k_i} \frac{1}{k_i^2} M_i.$$

Since

$$\sum_{i=1}^n \sum_{j=1}^{k_i} \frac{1}{k_i^2} M_i = \sum_{i=1}^n \frac{1}{k_i} M_i = \sum_{i=1}^n w_i^{-2} M_i,$$

we conclude that $A'^T A' = \tilde{A}^T \tilde{A}$, which implies $\left\|\tilde{A}\right\| = \|A'\|$.

Now consider the case of arbitrary $l_j \in \mathbb{N}$. Denote by B the $n \times (l_1 + \ldots + l_m)$ matrix, obtained from A by splitting each of its columns $a_{\cdot j}$ into l_j columns. Then $A \longrightarrow B \longrightarrow A'$.

By the previous arguments, $\|A'\| = \left\|\tilde{B}\right\|$, where $\tilde{B}$ is $n \times (l_1 + \ldots + l_m)$ matrix with ith row equal to

$$\left(\underbrace{\frac{a_{i1}}{l_1\sqrt{k_i}}}_{\text{repeated } l_1 \text{ times}} \quad \underbrace{\frac{a_{i2}}{l_2\sqrt{k_i}}}_{\text{repeated } l_2 \text{ times}} \quad \cdots \quad \underbrace{\frac{a_{im}}{l_m\sqrt{k_i}}}_{\text{repeated } l_m \text{ times}} \right).$$

Then the transpose of $\tilde{B}$ can be obtained from the $m \times n$ matrix $C = (C_{ji})$,

$$C_{ji} = \frac{a_{ji}}{\sqrt{k_i}}, \quad i \in [n],\ j \in [m],$$

by splitting the jth row of C into l_j rows.

By previous argument, $\left\|\tilde{B}^T\right\| = \left\|\tilde{C}\right\|$, where $\tilde{C} = \tilde{A}^T$. Thus we conclude

$$\|A'\| = \left\|\tilde{B}\right\| = \left\|\tilde{B}^T\right\| = \left\|\tilde{A}^T\right\| = \left\|\tilde{A}\right\|. \quad \blacktriangleleft$$

This shows that $\Gamma(A')$, for every matrix A' which can be obtained from A by splitting rows/columns, can be characterized by vectors w, v (s.t. the squares of components of w, v are rational numbers). The converse is also true:

▶ **Lemma A.4.** *Suppose that $A \in \mathcal{A}_{n,m}$ but vectors $w \in \mathbb{R}^n_+$, $v \in \mathbb{R}^m_+$ are such that $w_i^2 \in \mathbb{Q}$, $v_j^2 \in \mathbb{Q}$ for all i, j. Then there exist numbers $k_i \in \mathbb{N}$ and $l_j \in \mathbb{N}$ such that splitting A's ith row $a_{i\cdot}$ into k_i rows and the jth column $a_{\cdot j}$ into l_j rows yields a matrix A' such that $\Gamma(A') = \left\|\tilde{A}\right\| \|w\| \|v\|$ where $\left\|\tilde{A}\right\| = (\tilde{a}_{ij})$, $\tilde{a}_{ij} := \frac{a_{ij}}{w_i v_j}$.*

Proof. First note that the statement is true if $w_i^2 \in \mathbb{N}$ and $v_j^2 \in \mathbb{N}$ for all i, j, since then one takes $k_i = w_i^2$ and $l_j = v_j^2$.

Since $w_i^2 \in \mathbb{Q}$, $v_j^2 \in \mathbb{Q}$, we have $w_i^2 = \frac{p_i}{p'_i}$ and $v_j^2 = \frac{q_j}{q'_j}$ for some natural numbers p_i, p'_i, q_j and q'_j. Denote $P = \prod_i p'_i$ and $Q = \prod_j q'_j$. Let $\hat{w}_i = w_i\sqrt{P}$, $\hat{v}_j = v_j\sqrt{Q}$ and $\hat{A} = (\hat{a}_{ij})$, where $\hat{a}_{ij} = a_{ij}/(\hat{w}_i\hat{v}_j) = \tilde{a}_{ij}/\sqrt{PQ}$. Then

$$\left\|\hat{A}\right\| = \frac{1}{\sqrt{PQ}}\left\|\tilde{A}\right\|, \quad \|\hat{w}\| = \sqrt{P}\,\|w\|, \quad \|\hat{v}\| = \sqrt{Q}\,\|v\|, \quad \left\|\tilde{A}\right\| \|w\| \|v\| = \left\|\hat{A}\right\| \|\hat{w}\| \|\hat{v}\|.$$

Moreover, $\hat{w}_i^2 \in \mathbb{N}$, $\hat{v}_j^2 \in \mathbb{N}$, thus one can take $k_i = \hat{w}_i^2$ and $l_j = \hat{v}_j^2$. Now, by performing the corresponding row/column splitting, one obtains a matrix A' satisfying

$$\Gamma(A') = \left\|\hat{A}\right\| \|\hat{w}\| \|\hat{v}\| = \left\|\tilde{A}\right\| \|w\| \|v\|. \quad \blacktriangleleft$$

We can consider even more general situation:

▶ **Lemma A.5.** *Suppose that $A \in \mathcal{A}_{n,m}$ and $w \in \mathbb{R}^n_+$, $v \in \mathbb{R}^m_+$.*

Then there exist sequences $(k_{i,N})_N \subset \mathbb{N}$ and $(l_{j,N})_N \subset \mathbb{N}$ such that

$$\lim_{N\to\infty} \Gamma(A'_N) = \left\|\tilde{A}\right\| \|w\| \|v\|.$$

Here by $\tilde{A}$ we denote the matrix with components $\tilde{a}_{ij} = \frac{a_{ij}}{w_i v_j}$, but A'_N stands for the matrix which is obtained from A by splitting its ith row $a_{i\cdot}$ into $k_{i,N}$ rows and the jth column $a_{\cdot j}$ into $l_{j,N}$ rows.

Proof. We choose two sequences of vectors $w^{(1)}, w^{(2)}, \ldots$ and $v^{(1)}, v^{(2)}, \ldots$ so that $w^{(N)} \in Q^n_+$ and $w = \lim_{N\to\infty} w^{(N)}$ and similarly for $v^{(N)}$ and v. Let $\tilde{A}^{(N)}$ be a matrix with entries $\tilde{a}^{(N)}_{ij} = \frac{a_{ij}}{w_i v_j}$.

Then by Lemma A.4, there are matrices A'_N such that $\Gamma(A'_N) = \|\tilde{A}^{(N)}\|\|w^{(N)}\|\|v^{(N)}\|$. Let $k_{i,N}$ and $l_{i,N}$ be the values of k_i and l_i in the application of Lemma A.4. By continuity, if $N \to \infty$, we have $\|w^{(N)}\| \to \|w\|$, $\|v^{(N)}\| \to \|v\|$, $\|\tilde{A}^{(N)}\| \to \|\tilde{A}\|$.

Hence, $\lim_{N\to\infty} \Gamma(A'_N) = \left\|\tilde{A}\right\| \|w\| \|v\|$. ◀

Suppose that $A \in \mathcal{A}_{n,m}$ and $w \in \mathbb{R}^n_+$, $v \in \mathbb{R}^m_+$ are fixed. Let $\tilde{A}$ be the matrix with components $\tilde{a}_{ij} = a_{ij}/(w_i v_j)$. Notice that $\tilde{A} = D_w^{-1} A D_v^{-1}$. Denote $F_A(w,v) = \left\|D_w^{-1} A D_v^{-1}\right\| \|w\| \|v\|$. Then Claims A.3 and A.5 together imply that

$$\inf_{A' : A \longrightarrow A'} \Gamma(A') = \inf_{\substack{w \in \mathbb{R}^n_+ \\ v \in \mathbb{R}^m_+}} F_A(w,v).$$

Denote the latter infimum by F_A^T. In view of Lemma A.2 this means that

$$G(A) = \frac{\inf_{A' : A \longrightarrow A'} \Gamma(A')}{\|A\|_{\infty\to 1}} = \frac{F_A^T}{\|A\|_{\infty\to 1}}. \tag{6}$$

A.4 Proof of Lemma A.1

We recall the following characterization of matrices with $\|A\|_G \le 1$; for a proof, see [23, p. 239].

▶ **Lemma A.6.** *For every matrix A (of size $n \times n$), the inequality $\|A\|_G \le 1$ holds iff there is a matrix $\tilde{A}$ (of size $n \times n$) and vectors $w, v \in \mathbb{R}^n$ with non-negative components s.t. $\|w\| = \|v\| = 1$, $\left\|\tilde{A}\right\| \le 1$ and for all $i, j \in [n]$: $a_{ij} = \tilde{a}_{ij} w_i v_j$.*

From this it is easy to obtain the following:

▶ **Lemma A.7.** *For every matrix $A \in \mathcal{A}_{n,n}$ there exists a matrix $\tilde{A} \in \mathcal{A}_{n,n}$ and vectors $w, v \in \mathbb{R}^n_+$ s.t. $\|w\| = \|v\| = 1$, $\left\|\tilde{A}\right\| = \|A\|_G$ and $\tilde{A} = D_w^{-1} A D_v^{-1}$. Moreover, w and v minimize the function $F_A(\cdot,\cdot)$, i.e.,*

$$F_A^T = \left\|\tilde{A}\right\| \|w\| \|v\| = \|A\|_G .$$

Proof. Suppose that a matrix $A \in \mathcal{A}_{n,n}$ is scaled so that $\|A\|_G = 1$.

From Lemma A.6 the existence of $\tilde{A}$ with $\left\|\tilde{A}\right\| \le 1$ and $w, v \in \mathbb{R}^n_+$ with $\|w\| = \|v\| = 1$ follows. Notice that $w_i \neq 0$ and $v_j \neq 0$ for all i, j, since otherwise $A \notin \mathcal{A}$. Similarly, also $\tilde{A} \in \mathcal{A}_{n,n}$ must hold.

We claim that $\left\|\tilde{A}\right\| = 1$. Assume the contrary, $\left\|\tilde{A}\right\| = c \in (0,1)$.

Let $\tilde{B}$ be an $n \times n$ matrix with $\tilde{b}_{ij} = \tilde{a}_{ij}/c$, then $\left\|\tilde{B}\right\| = 1$ and by Lemma A.6 we have $\|B\|_G \le 1$, where $B = A/c$. But then $\|A\|_G \le c < 1$, a contradiction. Thus $\left\|\tilde{A}\right\|_G = 1$.

To prove the second part of the statement, suppose that there are unit vectors $\hat{w}, \hat{v} \in \mathbb{R}^n_+$ such that $F_A(\hat{w}, \hat{v}) = s < 1$. Let $\tilde{X} = D_{\hat{w}}^{-1} A D_{\hat{v}}^{-1}/s$, then $\left\|\tilde{X}\right\| = 1$. By Lemma A.6 we have $\|X\|_G \le 1$, where $X = A/s$. But then $\|A\|_G \le s < 1$, a contradiction. ◀

Proof of Lemma A.1.

The case of $A \in \mathcal{A}$. Notice that

$$\inf_{A' : A \longrightarrow A'} \Gamma(A') = \inf_{A' : A'' \longrightarrow A'} \Gamma(A'),$$

where A'' is any matrix s.t. $A \longrightarrow A''$. This means that $F_A^T = F_{A'}^T$, if $A \longrightarrow A'$. To apply Lemma A.7, transform A into a square matrix A' by splitting a row or a column. Then

$$F_A^T = F_{A'}^T \overset{\text{Lemma A.7}}{=} \|A'\|_G \overset{\text{Lemma A.2}}{=} \|A\|_G$$

and, by (6), $G(A) = \|A\|_G / \|A\|_{\infty\to 1}$, proving (5) for all $A \in \mathcal{A}$.

It remains to show that (5) holds for all matrices A.

The case of $A \notin \mathcal{A}$. Suppose that A is a $n \times m$ matrix and there are k zero rows and l zero columns. W.l.o.g. assume the non-zero rows/columns are the first, then

$$A = \begin{pmatrix} \hat{A} & 0_{n-k,l} \\ 0_{k,m-l} & 0_{k,l} \end{pmatrix},$$

where $\hat{A} \in \mathcal{A}_{n-k,m-l}$ (and $0_{a,b}$ stands for the zero matrix of size $a \times b$). Notice that

$$g(\hat{A}) = \frac{\left\|\hat{A}\right\| \sqrt{(n-k)(m-l)}}{\left\|\hat{A}\right\|_{\infty\to 1}} = \frac{\|A\| \sqrt{(n-k)(m-l)}}{\|A\|_{\infty\to 1}} < \frac{\|A\| \sqrt{nm}}{\|A\|_{\infty\to 1}} = g(A).$$

By the previous case, we have $G(\hat{A}) = \left\|\hat{A}\right\|_G / \left\|\hat{A}\right\|_{\infty\to 1} = \|A\|_G / \|A\|_{\infty\to 1}$.

Clearly, for every A' with $A \longrightarrow A'$ we have $\hat{A}'$ s.t. $\hat{A} \longrightarrow \hat{A}'$ and $g(\hat{A}') \leq g(A')$ (take $\hat{A}'$ to be the minor of A', obtained by skipping all zero rows or columns). Then $G(\hat{A}) \leq g(\hat{A}') < g(A')$. Taking infimum over all A' s.t. $A \longrightarrow A'$, inequality $G(\hat{A}) \leq G(A)$ follows.

On the other hand, for every $\hat{A}'$ s.t. $\hat{A} \longrightarrow \hat{A}'$ we have a sequence $(A_N)_{N\in\mathbb{N}}$ with $A \longrightarrow A_N$ for all N and $\lim_{N\to\infty} g(A_N) = g(\hat{A}')$: take the matrix

$$B = \begin{pmatrix} \hat{A}' & 0_{p,l} \\ 0_{k,q} & 0_{k,l} \end{pmatrix},$$

where $\hat{A}'$ is of size $p \times q$ (i.e., B is the matrix obtained by splitting the non-zero part of A in the same way how we split $\hat{A}$ to obtain $\hat{A}'$). Then the matrix A_N is obtained by splitting each row $b_{i\cdot}$, $i \in [p]$ of B, and each column $b_{\cdot j}$, $j \in [q]$ of B into N rows/columns. We have $A \longrightarrow B \longrightarrow A_N$ and the resulting matrix A_N is of size $(Np+k) \times (Nq+l)$. We denote the upper $Np \times Nq$ submatrix of A_N by B_N. Then $B_N = \frac{1}{N^2}\hat{A}' \otimes J_{N,N}$, where $J_{N,N}$ is the $N \times N$ all-1 matrix.

We have

$$\|A_N\| = \|B_N\| = \frac{\left\|\hat{A}'\right\|}{N};$$

$$\|A_N\|_{\infty\to 1} = \|B_N\|_{\infty\to 1} = \left\|\hat{A}'\right\|_{\infty\to 1};$$

$$g(A_N) = \frac{\|A_N\| \sqrt{(Np+k)\cdot(Nq+l)}}{\|A_N\|_{\infty\to 1}} = \frac{\|B_N\| \sqrt{(Np+k)\cdot(Nq+l)}}{\|B_N\|_{\infty\to 1}}$$

$$= \frac{\left\|\hat{A}'\right\| \sqrt{pq}}{\left\|\hat{A}'\right\|_{\infty\to 1}} \cdot \sqrt{\frac{Np+k}{Np} \cdot \frac{Nq+l}{Nq}} = g(\hat{A}') \sqrt{\left(1+\frac{c_1}{N}\right)\left(1+\frac{c_2}{N}\right)},$$

where $c_1 = k/p$, $c_2 = l/q$.

We see that $G(A) \leq \lim_{N\to\infty} g(A_N) = g(\hat{A}')$. Taking infimum over all $\hat{A}'$ s.t. $\hat{A} \longrightarrow \hat{A}'$, inequality $G(\hat{A}) \geq G(A)$ follows. Hence the two quantities must be equal. ◀

B Proof of Theorem 4.1

We use the notion of *certificate complexity*. Let C be an assignment of values $C : S \to \{0,1\}$ for some $S \subseteq [n]$. We say that $x = (x_1, \ldots, x_n)$ is *consistent* with C if it satisfies $x_i = C(i)$

for all $i \in S$. We say that C is a *certificate* for f on an input x if x is consistent with C and, for any $y \in \{0,1\}^n$ that is consistent with C, we have $f(y) = f(x)$.

The certificate complexity of f on an input x (denoted by $C(f,x)$) is the smallest $|S|$ in a certificate C for f on the input x. The certificate complexity of f (denoted $C(x)$) is the maximum of $C(f,x)$ over all $x \in \{0,1\}^n$. (For more information on the certificate complexity and its connections to other complexity measures, we refer the reader to the survey by Buhrman and de Wolf [13].)

We use the same function as in the $Q_\epsilon(f) = \tilde{\Omega}(\deg^2(f))$ result of Aaronson et al. [3]. The construction of this function [3] starts by designing a function $g : \{-1,1\}^n \to \{0,1\}$ with $Q_\epsilon(g) = \tilde{\Omega}(n)$ and $C(g) = \tilde{O}(\sqrt{n})$. (We omit the definition of g because $Q_\epsilon(g) = \tilde{\Omega}(n)$ and $C(g) = \tilde{O}(\sqrt{n})$ are the only properties of g that we use.)

Then they define f as follows:

1. The first $c = 10n \log n$ input variables of f are interpreted as c inputs $x^{(1)} \in \{0,1\}^n$, ..., $x^{(c)} \in \{0,1\}^n$ to the function g.
2. These input variables are followed by 2^c groups of variables $y^{(m)}$, $m \in \{0,1\}^c$, with each group containing $cC(g)\log n$ variables. The content of each $y^{(m)}$ is interpreted as descriptions for c sets $S_1, \ldots, S_c \subseteq [n]$ with $|S_j| = C(g)$. A set S_j is interpreted as a sequence of indices for $C(g)$ variables for the function $g(x^{(j)})$.
3. $f = 1$ if and only if, for some $m \in \{0,1\}^c$, the group $y^{(m)}$ contains descriptions for sets S_i such that, for each $i \in [c]$, the variables $x_j^{(i)}$, $j \in S_i$ form an m_i-certificate.

As shown in [3], f satisfies $Q_\epsilon(f) = \tilde{\Omega}(n)$ and $\deg(f) = \tilde{O}(\sqrt{n})$. A polynomial p of degree $\tilde{O}(\sqrt{n})$ that represents f can be constructed as follows:

1. $p = \sum_{m \in \{0,1\}^c} p_m$;
2. $p_m = \sum_{S_1,\ldots,S_c} p_{m,S_1,\ldots,S_c}$, with the summation over all tuples $(S_1, \ldots, S_c)$ such that, for all $i \in [c]$, S_i is a possible certificate for $g(x) = m_i$;
3. $p_{m,S_1,\ldots,S_c} = q_{m,S_1,\ldots,S_c} \prod_{i=1}^c r_{i,m_i,S_i}$;
4. $q_{m,S_1,\ldots,S_c} = 1$ if the contents of $y^{(m)}$ describe sets $S_1, \ldots, S_c$ and $q_{m,S_1,\ldots,S_c} = 0$ otherwise;
5. $r_{i,m_i,S_i} = 1$ if the values of variables $x_j^{(i)}, j \in S_i$ certify that $g(x^{(i)}) = m_i$ and $r_{i,m_i,S_i} = 0$ otherwise.

In the non-block-multilinear case, $q_{m,S_1,\ldots,S_c}$ is the product of $\frac{1+y_i^{(m)}}{2}$'s (for i's where we need $y_i^{(m)} = 1$) and $\frac{1-y_i^{(m)}}{2}$'s (for i's where we need $y_i^{(m)} = -1$). r_{i,m_i,S_i} is constructed similarly, by taking a product of $\frac{1+x_j^{(i)}}{2}$'s and $\frac{1-x_j^{(i)}}{2}$'s for $j \in S_i$, to obtain the condition that $x_j^{(i)}$ take the values that are necessary so that $x_j^{(i)}$, $j \in S_i$, certify $g(x^{(i)}) = m_i$.

We now modify this construction to obtain $\mathrm{bmdeg}(f) = \tilde{O}(\sqrt{n})$. Our polynomial has blocks of variables $z^{(i)}$, for $i \in [cC(g)(\log n + 1)]$, with each $z^{(i)}$ consisting of a variable $z_0^{(i)}$, c subblocks $x^{(i,1)}, \ldots, x^{(i,c)}$ and 2^c subblocks $y^{(i,m)}$ for $m \in \{0,1\}^c$.

The structure of the polynomial p stays the same and we only modify the constructions of $q_{m,S_1,\ldots,S_c}$ and r_{i,m_i,S_i}. To construct $q_{m,S_1,\ldots,S_c}$, we use the first $cC(g)\log n$ blocks $z^{(i)}$, taking the value of $y_i^{(m)}$ from the i^{th} block and using $z_0^{(i)}$ instead of 1 in the terms $\frac{1 \pm y_i^{(m)}}{2}$.

To construct r_{i,m_i,S_i}, we use $z^{(k)}$ for $k \in \{(c\log n + (i-1))C(g)+1, \ldots, (c\log n + i)C(g)\}$ and take r_{i,m_i,S_i} to be the average of the desired product of $\frac{z_0^{(k)} + x_j^{(k,i)}}{2}$'s and $\frac{z_0^{(k)} - x_j^{(k,i)}}{2}$'s over all the ways how one could use one term per block $z^{(k)}$.

It is easy to see that, if all blocks $z^{(i)}$ contain the same assignment z, then $p(z, \ldots, z)$ is the same polynomial as in the non-block-multilinear case and is equal to $f(z)$. We now show that $|p| \leq 1$ for any choice of $z^{(1)}, z^{(2)}, \ldots$ in which all the variables are in $\{-1, 1\}$.

For each m, all polynomials $q_{m,S_1,\ldots,S_c}$ use the same variables $z_0^{(i)}$ and $y_i^{(i,m)}$ and are defined so that, for any choice of values for $z_0^{(i)}$'s and $y_i^{(i,m)}$'s, at most one of $q_{m,S_1,\ldots,S_c}$ is ± 1 and the rest are 0. Let $S_{m,1},\ldots,S_{m,c}$ be the sets for which $q_{m,S_{m,1},\ldots,S_{m,c}} = \pm 1$ (if such sets exist). Then $p(z^{(1)},\ldots,z^{(cC(g)(\log n+1))})$ is equal to the sum

$$\sum_{m\in\{0,1\}^c} a_m \prod_{i=1}^{c} r_{i,m_i,S_{m,i}} \tag{7}$$

for some choice of signs $a_m \in \{-1,1\}$. We show

▶ **Lemma B.1.** *Let $S_{m,i}$, $m \in \{0,1\}^c$, $i \in [c]$ be such that $S_{m,i}$ is an m_i-certificate for the function g. Then*

$$\left| \sum_{m\in\{0,1\}^c} a_m \prod_{i=1}^{c} r_{i,m_i,S_{m,i}} \right| \leq 1$$

for any choice of signs $a_m \in \{-1,1\}$.

Proof. By induction on c. For $c = 1$, this simplifies to

$$-1 \leq a_0 r_{1,0,S_{0,1}} + a_1 r_{1,1,S_{1,1}} \leq 1 \tag{8}$$

when $S_{0,1}$ is a set of variables for a 0-certificate and $S_{1,1}$ is a set of variables for a 1-certificate. Since a 0-certificate and a 1-certificate cannot be true at the same time, there must be $j \in S_{0,1} \cap S_{1,1}$ with x_j taking one value in the 0-certificate and another value in the 1-certificate.

Let p_0 be the probability that, when we choose a block $z^{(i)}$ randomly among the blocks that are used to define r_{1,m_1,S_1}'s, we get the value of $x_j^{(i,1)}$ which matches the 0-certificate. Then the probability of getting the value that matches the 1-certificate is $1 - p_0$ and we get that $r_{1,0,S_{0,1}} \leq p_0$ and $r_{1,1,S_{1,1}} \leq 1 - p_0$. This implies (8) for any choice of signs $a_0, a_1 \in \{-1,1\}$.

For $c > 1$, we can use the same argument to show that, for any $m \in \{0,1\}^{c-1}$, we have $r_{c,0,S_{m0}} \leq p_m$ and $r_{c,1,S_{m1}} \leq 1 - p_m$ for some p_m that depends on m. Therefore, the sum of Lemma B.1 is upper bounded by

$$\sum_{m\in\{0,1\}^{c-1}} \left(p_m a_{m0} \prod_{i=1}^{c-1} r_{i,m_i,S_{m0,i}} + (1-p_m) a_{m1} \prod_{i=1}^{c-1} r_{i,m_i,S_{m1,i}} \right).$$

We can express this sum as a probabilistic combination of sums

$$\sum_{m\in\{0,1\}^{c-1}} a_m \prod_{i=1}^{c-1} r_{i,m_i,S_{m,i}} \tag{9}$$

where each $S_{m,i}$ is either $S_{m0,i}$ or $S_{m1,i}$ and each a_m is either a_{m0} or a_{m1}. Each of sums (9) is at most 1 in absolute value by the inductive assumption. ◀

Sculpting Quantum Speedups*

Scott Aaronson[1] and Shalev Ben-David[2]

1 **Massachusetts Institute of Technology, Cambridge, USA**
`aaronson@csail.mit.edu`
2 **Massachusetts Institute of Technology, Cambridge, USA**
`shalev@mit.edu`

Abstract

Given a problem which is intractable for both quantum and classical algorithms, can we find a sub-problem for which quantum algorithms provide an exponential advantage? We refer to this problem as the "sculpting problem." In this work, we give a full characterization of sculptable functions in the query complexity setting. We show that a total function f can be restricted to a promise P such that $\mathrm{Q}(f|_P) = O(\mathrm{polylog}\, N)$ and $\mathrm{R}(f|_P) = N^{\Omega(1)}$, if and only if f has a large number of inputs with large certificate complexity. The proof uses some interesting techniques: for one direction, we introduce new relationships between randomized and quantum query complexity in various settings, and for the other direction, we use a recent result from communication complexity due to Klartag and Regev. We also characterize sculpting for other query complexity measures, such as $\mathrm{R}(f)$ vs. $\mathrm{R}_0(f)$ and $\mathrm{R}_0(f)$ vs. $\mathrm{D}(f)$.

Along the way, we prove some new relationships for quantum query complexity: for example, a nearly quadratic relationship between $\mathrm{Q}(f)$ and $\mathrm{D}(f)$ whenever the promise of f is small. This contrasts with the recent super-quadratic query complexity separations, showing that the maximum gap between classical and quantum query complexities is indeed quadratic in various settings – just not for total functions!

Lastly, we investigate sculpting in the Turing machine model. We show that if there is any BPP-bi-immune language in BQP, then *every* language outside BPP can be restricted to a promise which places it in PromiseBQP but not in PromiseBPP. Under a weaker assumption, that some problem in BQP is hard on average for P/poly, we show that every *paddable* language outside BPP is sculptable in this way.

1998 ACM Subject Classification F.1.2 Modes of Computation

Keywords and phrases Quantum Computing, Query Complexity, Decision Tree Complexity, Structural Complexity

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.26

1 Introduction

When are quantum algorithms useful? In general, quantum algorithms are believed to provide exponential speedups for certain structured problems, such as factoring [18], but not for unstructured problems like NP-complete problems.

In this work, we ask the question in a new way. Given a problem for which quantum algorithms are not useful, can we nevertheless find a sub-problem on which they provide an exponential advantage over classical algorithms? We call this the "sculpting" question: our goal is to sculpt the original intractable problem into a sub-problem that's still classically

* This work was partially supported by an Alan T. Waterman Award from the National Science Foundation, under grant no. 1249349.

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 26; pp. 26:1–26:28

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

intractable, but for which there exists a fast quantum algorithm. The sculpting question arises, for example, in adiabatic quantum computation: while it is not believed that adiabatic quantum computing can solve NP-complete problems in polynomial time, a widely discussed question is whether there is a sub-problem of SAT on which adiabatic computing provides an exponential advantage over classical algorithms.

We study the sculpting question primarily in the query complexity model. The utility of the model comes from its relative tractability: for example, in query complexity, Shor's period finding algorithm provides a provable exponential speedup over any classical algorithm [8].

In query complexity, we're given a (possibly partial) function $f : \{0,1\}^N \to \{0,1\}$ and an oracle access to a string $x \in \{0,1\}^N$. The goal is to evaluate $f(x)$ using as few oracle calls to the entries of x as possible. The minimum number of queries required by an algorithm for computing $f(x)$ (over the worst-case choice of x) is the query complexity of f. If the algorithm in question is deterministic, we denote this by $\mathrm{D}(f)$; if it is zero-error randomized, we denote this by $\mathrm{R}_0(f)$; if it is (bounded error) randomized, we denote this by $\mathrm{R}(f)$; and if it is (bounded error) quantum, we denote it by $\mathrm{Q}(f)$.

In this query complexity setting, the sculpting question can be phrased as follows: given a total function $f : \{0,1\}^N \to \{0,1\}$ for which $\mathrm{R}(f)$ and $\mathrm{Q}(f)$ are both large (say, $N^{\Omega(1)}$), is there a promise $P \subseteq \{0,1\}^N$ such that $f|_P$, the restriction of f to P, has $\mathrm{Q}(f|_P) = O(\operatorname{polylog} N)$ and $\mathrm{R}(f|_P) = N^{\Omega(1)}$?

For example, if f is the OR function, such sculpting is not possible, as follows from [1]. As another example, if f is defined to be 1 when Simon's condition is satisfied and 0 otherwise, then sculpting is possible: the promise will simply restrict to inputs that either satisfy Simon's condition or are far from satisfying it; this promise suffices for an exponential quantum speedup [7].

We fully characterize the functions f for which such a promise exists. In particular, we show that sufficiently "rich" functions, such as Parity or Majority, are sculptable.

The sculpting problem has been previously studied by Zhan, Kimmel, and Hassidim [20] for the case where f a recursive function such as the NAND-tree. They constructed a promise on which this function gives a small super-polynomial speedup ($\operatorname{polylog}(n)$ vs. $(\log n)^{\Omega(\log\log\log n)}$). Our results also apply to such recursive functions, and we improve the speedup to $\operatorname{polylog} n$ vs. $n^{\Omega(1)}$.

Our sculpting construction uses communication complexity in a novel way. In the other direction, to prove non-sculptability, we prove new query complexity relationships. As a corollary, we get nearly quadratic relationships between classical and quantum query complexities for a wider class of functions than previously known.

Results

H-indices

We introduce a new query complexity measure, $\mathrm{H}(\mathrm{C}_f)$, defined as the maximum number h for which there are 2^h inputs to f with certificate complexity at least h. We call this the H-index of certificate complexity (motivated by the citation H-index sometimes used to measure research productivity [10]). This quantity measures the number of inputs there are to a function f that have large certificate complexity. We prove various properties of $\mathrm{H}(\mathrm{C}_f)$; most notably, we show that for total functions, it is nearly quadratically related to $\mathrm{H}(\mathrm{bs}_f)$, the H-index of block sensitivity. This is analogous to the quadratic relationship between C and bs.

Sculpting in Query Complexity

Our main result is the following theorem, which neatly characterizes sculptability in the query complexity model in terms of the H-index of certificate complexity.

▶ **Theorem 1.1.** *Let $f : \{0,1\}^N \to \{0,1\}$ be a total function. Then there is a promise $P \subseteq \{0,1\}^N$ such that $\mathrm{R}(f|_P) = N^{\Omega(1)}$ and $\mathrm{Q}(f|_P) = N^{o(1)}$, if and only if $\mathrm{H}(\mathrm{C}_f) = N^{\Omega(1)}$. Furthermore, in this case we also have $\mathrm{Q}(f|_P) = O(\log^2 N)$.*

This theorem follows as an immediate corollary of the following more general characterization theorem.

▶ **Theorem 1.2.** *For all total functions $f : \{0,1\}^N \to \{0,1\}$ and all promises $P \subseteq \{0,1\}^N$, we have*

$$\mathrm{R}(f|_P) = O(\mathrm{Q}(f|_P)^2 \,\mathrm{H}(\mathrm{C}_f)^2).$$

Conversely, for all total functions $f : \{0,1\}^N \to \{0,1\}$, there is a promise $P \subseteq \{0,1\}^N$ such that

$$\mathrm{R}(f|_P) = \Omega\left(\frac{\mathrm{H}(\mathrm{C}_f)^{1/6}}{\log^{11/6} N}\right) \qquad \text{and} \qquad \mathrm{Q}(f|_P) = O(\log^2 \mathrm{H}(\mathrm{C}_f)).$$

We also prove an analogous theorem for D vs. R_0, showing that the same $\mathrm{H}(\mathrm{C}_f) = N^{\Omega(1)}$ condition also characterizes sculpting $\mathrm{D}(f)$ vs. $\mathrm{R}_0(f)$. On the other hand, we show that sculpting $\mathrm{R}_0(f)$ vs. $\mathrm{R}(f)$ is *always* possible: for every total function f with $\mathrm{R}_0(f) = N^{\Omega(1)}$, there is a promise P such that $\mathrm{R}_0(f|_P) = N^{\Omega(1)}$ and $\mathrm{R}(f|_P) = O(1)$.

Query Complexity on Small Promises

On the way to proving Theorem 1.2, we prove the following theorem, providing a quadratic relationship between $\mathrm{Q}(f)$ and $\mathrm{D}(f)$ when the domain of f is small. This provides a ironic twist to the query complexity story: for a long time, it was believed that $\mathrm{D}(f)$ and $\mathrm{Q}(f)$ are quadratically related when the domain of f is very large (in particular, for total functions). This conjecture was recently disproven by [3] (who showed a $\mathrm{D}(f) \sim \mathrm{Q}(f)^4$ separation) and by [2] (who showed an f such that $\mathrm{R}(f) \sim \mathrm{Q}(f)^{2.5}$). Instead, we now show that the quadratic relationship holds when the domain of f is very *small*.

▶ **Theorem 1.3.** *Let $f : \{0,1\}^N \to \{0,1\}$ be a partial function, and let $\mathrm{Dom}(f)$ denote the domain of f. Then*

$$\mathrm{Q}(f) = \Omega\left(\frac{\sqrt{\mathrm{D}(f)}}{\log|\mathrm{Dom}(f)|}\right).$$

Query Complexity for Unbalanced Functions

We show two relationships similar to Theorem 1.3 that hold for functions whose domain is large, but which are unbalanced: they contain very few 0-inputs compared to 1-inputs, or vice versa.

▶ **Theorem 1.4.** *Let $f : \{0,1\}^N \to \{0,1\}$ be a partial function. Define the measure $\mathrm{Bal}(f) \in [0, N]$ as $\mathrm{Bal}(f) := 1 + \min\{\log|f^{-1}(0)|, \log|f^{-1}(1)|\}$ (or 0 if f is constant). Then*

$$\mathrm{R}(f) = O(\mathrm{Q}(f)^2 \,\mathrm{Bal}(f))$$

$$\mathrm{D}(f) = O(\mathrm{R}_0(f) \,\mathrm{Bal}(f)).$$

A similar polynomial relationship between $\mathrm{R}_0(f)$ and $\mathrm{R}(f)$ does not hold in general.

New Relationship for Total Functions

We prove the following new query complexity relationship for total functions, generalizing the known relationship $\mathrm{D}(f) = O(\mathrm{Q}(f)^2\,\mathrm{C}(f))$.

▶ **Theorem 1.5.** *Let $f : \{0,1\}^N \to \{0,1\}$ be a total function. Then*

$$\mathrm{D}(f) = O(\mathrm{Q}(f)^2\,\mathrm{H}(\sqrt{\mathrm{C}_f})^2).$$

Here $\mathrm{H}(\sqrt{\mathrm{C}_f})$ denotes the H-index of the square root of certificate complexity; this is the maximum number h such that there are at least 2^h inputs to f for which $\sqrt{\mathrm{C}(f)}$ is at least h. We note that $\mathrm{H}(\sqrt{\mathrm{C}_f})^2 \leq \mathrm{C}(f)$ for all total functions, so this is an improvement over the relationship $\mathrm{D}(f) = O(\mathrm{Q}(f)^2\,\mathrm{C}(f))$. Moreover, when $f = \mathrm{OR}$, we have $\mathrm{H}(\sqrt{\mathrm{C}_f})^2 = 1$ and $\mathrm{C}(f) = N$, so this improvement is strict.

We remark that this result could let us improve the relationship $\mathrm{D}(f) = O(\mathrm{Q}(f)^6)$ if we could show $\mathrm{H}(\sqrt{\mathrm{C}_f})^2 = o(\mathrm{Q}(f)^4)$. Theorem 1.5 therefore provides a new approach for this long-standing open problem.

Sculpting in the Turing Machine Model

In Section 8, we examine sculpting in the Turing machine model. We say that a language L is *sculptable* if there is a promise set P such that the promise problem of deciding if an input from P is in L is in PromiseBQP but not in PromiseBPP. We prove two sculptability theorems, both of them providing evidence that most or all languages outside of BPP are sculptable.

▶ **Theorem 1.6.** *Assume* PromiseBQP *is hard on average for* P/poly*. Then every paddable language outside of* BPP *is sculptable.*

▶ **Theorem 1.7.** *Assume there exists a* BPP*-bi-immune language in* BQP*. Then every language outside of* BPP *is sculptable.*

For the definitions of paddability and bi-immunity, see Section 8. These theorems assume very little about BQP and BPP, and analogous statements hold for other pairs of complexity classes.

2 Preliminaries

For a (possibly partial) function $f : \{0,1\}^N \to \{0,1\}$, we use $\mathrm{D}(f)$, $\mathrm{R}_0(f)$, $\mathrm{R}(f)$, and $\mathrm{Q}(f)$ to denote the deterministic query complexity, zero-error randomized query complexity, bounded-error randomized query complexity, and bounded-error quantum query complexity of f, respectively. For the definitions of these measures, see [6].

A partial assignment is a string p in $\{0,1,*\}^N$ that represents partial knowledge of a string in $\{0,1\}^N$. For $x \in \{0,1\}^N$, we say that p is a partial assignment of x if x extends p; that is, if x and p agree on all the non-$*$ entries of p. A partial assignment p for x is called a *certificate* for x if all strings that extend p have that same value under f as x; that is, if for all $y \in \{0,1\}^N$ that extend p, we have $f(y) = f(x)$. The certificate complexity of f on input x, denoted by $\mathrm{C}_f(x)$, is the minimum number of bits in any certificate of x with respect to f. The certificate complexity of f, denoted by $\mathrm{C}(f)$, is defined as the maximum of $\mathrm{C}_f(x)$ over all strings x in the domain of f.

The certificate complexity $\mathrm{C}_f(x)$ can be thought of as the deterministic query complexity of f given the promise that the input is either x or else an input y such that $f(x) \neq f(y)$. Motivated by this observation, Aaronson [1] defined the *randomized certificate complexity* of x, denoted by $\mathrm{RC}_f(x)$, to be the (bounded-error) randomized query complexity of f on this promise. He defined the quantum certificate complexity $\mathrm{QC}_f(x)$ analogously. As with C, we use $\mathrm{RC}(f)$ to denote the maximum of $\mathrm{RC}_f(x)$ over all x in the domain of f, and define $\mathrm{QC}(f)$ similarly.

For any string $x \in \{0,1\}^N$ and set of bits $B \subseteq \{1,2,\ldots,N\}$, denote by x^B the string x with the bits in B flipped. For any $f : \{0,1\}^N \to \{0,1\}$, if $f(x) \neq f(x^B)$, we say that B is a *sensitive block* for x with respect to f. The *block sensitivity* of x, denoted by $\mathrm{bs}_f(x)$, is the maximum number of disjoint sensitive blocks for x. Note that the block sensitivity is the packing number of the collection of sensitive blocks of x. It can be seen that $\mathrm{C}_f(x)$ can be interpreted as the hitting number of that collection (the minimum number of bits in a set that has non-empty intersection with all the blocks).

The linear programming relaxation of the packing problem is the dual of the linear programming relaxation of the hitting problem. We call the optimum of this LP the fractional block sensitivity (or fractional certificate complexity). As it happens, this measure is equal to $\mathrm{RC}_f(x)$. This observation is implicit in [1], and was made explicit in [13].

Actually, the fractional block sensitivity differs by a constant factor from Aaronson's original definition of RC. In this work we will use RC to denote the fractional block sensitivity. Another property of RC that we will need is that $1/\mathrm{RC}_f(x)$ is equal to the minimum infinity-norm distance between x and the convex hull of the set of inputs y such that $f(y) \neq f(x)$. That is, if $f(x) = 0$ and $S = f^{-1}(1)$, we have

$$\frac{1}{\mathrm{RC}_f(x)} = \min_{\mu \in \Delta_S} \max_i \Pr_{y \sim \mu}[x_i \neq y_i],$$

where Δ_S is the set of all probability distributions over S (equivalently, the convex hull of S). In particular, this minimum is attained, so there is a probability distribution μ over $f^{-1}(1)$ such that for all $i = 1, 2, \ldots, n$, if we sample $y \sim \mu$ we get $\Pr[y_i \neq x_i] \leq 1/\mathrm{RC}_f(x)$.

Clearly, for all $f : \{0,1\}^N \to \{0,1\}$ we have

$$\{\mathrm{QC}(f), \mathrm{bs}(f)\} \leq \mathrm{RC}(f) \leq \{\mathrm{C}(f), \mathrm{R}(f)\} \leq \mathrm{R}_0(f) \leq \mathrm{D}(f),$$

with $\mathrm{Q}(f)$ lying between $\mathrm{QC}(f)$ and $\mathrm{R}(f)$. Aaronson [1] showed that $\mathrm{RC}_f(x) = \Theta(\mathrm{QC}_f(x)^2)$ for all f and x, so in particular, $\mathrm{RC}(f) = \Theta(\mathrm{QC}(f)^2)$. In addition, when f is total, we can relate all these measures to each other: we have

$$\mathrm{D}(f) = O(\mathrm{bs}(f)^3) = O(\mathrm{RC}(f)^3) = O(\mathrm{QC}(f)^6) = O(\mathrm{Q}(f)^6),$$

with the first equality following from [4].

Balance and H Indices

We will use $\mathrm{Dom}(f)$ to denote the domain of a partial function f. We define $\mathrm{Bal}(f)$ to be 0 if f is constant, and otherwise, to be the minimum of $1 + \log|f^{-1}(0)|$ and $1 + \log|f^{-1}(1)|$ (we use log to denote logarithm base 2). Note that since $|f^{-1}(0)| + |f^{-1}(1)| = |\mathrm{Dom}(f)| \leq 2^N$, we have $\mathrm{Bal}(f) \leq N$. Thus $\mathrm{Bal}(f) \in [0, N]$.

We will use a new set of query complexity measures called H-indices (the name is motivated by the H-index measure of citations, a common metric for research output). For a given function $g : \{0,1\}^N \to [0, \infty)$, we will define the H-index of g, denoted by $\mathrm{H}(g)$, as the

maximum number h such that there are at least 2^h inputs with $g(x) \geq h$. Alternatively, the H-index of g can be defined as the minimum number h such that there are at most 2^h inputs with $g(x) > h$. It is not obvious that these definitions are equivalent (or even that the minimum and maximum are attained); we prove this in Appendix A.

Note that $\mathrm{H}(g) \in [0, N]$, and $\mathrm{H}(g) \leq \max_x g(x)$. Also, if $g(x) \geq g'(x)$ for all $x \in \{0,1\}^N$, we have $\mathrm{H}(g) \geq \mathrm{H}(g')$.

We'll primarily be interested in measures like $\mathrm{H}(\mathrm{C}_f)$, $\mathrm{H}(\mathrm{RC}_f)$, and $\mathrm{H}(\mathrm{bs}_f)$. We have $\mathrm{H}(\mathrm{C}_f) \leq \mathrm{C}(f)$, $\mathrm{H}(\mathrm{RC}_f) \leq \mathrm{RC}(f)$, and $\mathrm{H}(\mathrm{bs}_f) \leq \mathrm{bs}(f)$. We also have

$$\mathrm{H}(\mathrm{bs}_f) \leq \mathrm{H}(\mathrm{RC}_f) \leq \mathrm{H}(C_f).$$

The H-index of certificate complexity can be much smaller than the certificate complexity itself. For example, the OR function has only one certificate of size greater than 1, so $\mathrm{H}(\mathrm{C}_{\mathrm{OR}}) = 1$, even though $\mathrm{C}(\mathrm{OR}) = n$.

In Appendix A we show that if $\alpha : [0,\infty) \to [0,\infty)$ is an increasing function, then

$$\mathrm{H}(\alpha \circ g) \leq \max\{\mathrm{H}(g), \alpha(\mathrm{H}(g))\}.$$

In particular, this will imply $\mathrm{H}(\mathrm{C}_f^2) \leq \mathrm{H}(\mathrm{C}_f)^2$.

Shattering and the Sauer-Shelah Lemma

For a set of indices $A \subseteq \{1, 2, \ldots, N\}$, let $S|_A \subseteq \{0,1\}^{|A|}$ be the set of restrictions of each string in S to the indices in A. We say A is shattered by S if $S|_A = \{0,1\}^{|A|}$. In other words, A is shattered by S if S has all possible behaviors on A. The Sauer-Shelah lemma [16, 17] is a classic result that upper-bounds the size of S in terms of the size of A. We will use the following corollary of it.

▶ **Lemma 2.1.** *Let $S \subseteq \{0,1\}^N$ be a collection of strings. Then there is a shattered set of indices of size at least*

$$\frac{\log |S|}{\log(N+1)}.$$

Lemma 2.1 follows straightforwardly from the Sauer-Shelah lemma, as we prove in Appendix B. We will often use the weaker bound $\frac{\log |S|}{2 \log N}$ instead, which holds for $N \geq 2$. This will sometimes lead to simpler formulas.

3 Non-Sculptability Theorems

In this section, we prove the non-sculptability direction of Theorem 1.2. The proof has two parts: in Section 3.1, we prove a relationship between randomized and quantum query complexities for "unbalanced" functions, and in Section 3.2, we use this to prove a sculpting lower bound in terms of the H-index of certificate complexity.

3.1 Query Complexity for Unbalanced Functions

We wish to show a nearly-quadratic relationship between randomized and quantum query complexities for functions f for which $\mathrm{Bal}(f)$ is small. Note that this is a generalization of the relation $\mathrm{RC}_f(x) = O(\mathrm{Q}_f(x)^2)$ from [1]. That is, [1] showed that for the task of distinguishing one input from a (possibly large) set of alternatives, randomized and quantum algorithms

are quadratically related. We want a similar relationship for the task of distinguishing a *small set* of inputs from a (possibly large) set of alternatives.

We start with the following lemma.

▶ **Lemma 3.1.** *Let $f : \{0,1\}^N \to \{0,1\}$ be a partial function. For $a \notin f^{-1}(0)$, let $f_{a,0}$ be the problem of distinguishing a from $f^{-1}(0)$. That is, $f_{a,0}$ is the function $f_{a,0} : \{a\} \cup f^{-1}(0) \to \{0,1\}$ with $f(x) = 1$ iff $x = a$. For $a \notin f^{-1}(1)$, define $f_{a,1}$ analogously. Then for all $a \in \{0,1\}^N$, we have either $\mathrm{R}(f_{a,0}) = O(\mathrm{Q}(f)^2)$ or $\mathrm{R}(f_{a,1}) = O(\mathrm{Q}(f)^2)$.*

Note that this holds even when a is not in the promise of f. The constant in the big-O notation is a universal constant independent of a, f, and N.

Proof. Let Q be the quantum algorithm that achieves $\mathrm{Q}(f)$ quantum query complexity in determining the value of f on a given input. When run on any $a \in f^{-1}(0)$, Q will output 0 with probability at least $2/3$, and when run on $a \in f^{-1}(1)$, it will output 1 with probability at least $2/3$.

Consider running Q on an input $a \notin \mathrm{Dom}(f)$. Then Q will output 0 with some probability p and output 1 with probability $1-p$. If $p \geq 1/2$, then Q distinguishes a from $f^{-1}(1)$ with constant probability. If $p \leq 1/2$, then Q distinguishes a from $f^{-1}(0)$ with constant probability. Thus for all $a \in \{0,1\}^N$, we have either $\mathrm{Q}(f_{a,0}) = O(\mathrm{Q}(f))$ or $\mathrm{Q}(f_{a,1}) = O(\mathrm{Q}(f))$. From [1], we have $\mathrm{RC}(g) = O(\mathrm{QC}(g)^2) = O(\mathrm{Q}(g)^2)$ for all functions g, so we conclude that either $\mathrm{RC}(f_{a,0}) = O(\mathrm{Q}(f)^2)$ or $\mathrm{RC}(f_{a,1}) = O(\mathrm{Q}(f)^2)$.

Finally, note that for a problem of distinguishing one input from the rest, randomized query complexity equals randomized certificate complexity. Thus we get that for all $a \in \{0,1\}^N$, either $\mathrm{R}(f_{a,0}) = O(\mathrm{Q}(f)^2)$ and or $\mathrm{R}(f_{a,1}) = O(\mathrm{Q}(f)^2)$. ◀

We're now ready to prove the desired relationship between R and Q.

▶ **Theorem 3.2.** *Let $f : \{0,1\}^N \to \{0,1\}$ be a partial function. Then*

$$\mathrm{R}(f) = O(\mathrm{Q}(f)^2 \,\mathrm{Bal}(f)).$$

Proof. Without loss of generality, assume $|f^{-1}(0)| \leq |f^{-1}(1)|$. We use Lemma 3.1 to construct a randomized algorithm for determining $f(x)$ given oracle access to x, assuming that $f^{-1}(0)$ is small. The idea is to keep track of the subset $Z \subseteq f^{-1}(0)$ of strings that the input x might feasibly be (consistent with the queries seen so far). We then construct a string a from a majority vote of the elements of Z; that is, for each index $i \in [n]$, a_i will be the majority of y_i over all $y \in Z$ (with ties broken arbitrarily).

This string a need not be in $\mathrm{Dom}(f)$. The important property of it is that if we query an index i of the input x and discover that $x_i \neq a_i$, we can eliminate at least half of the strings from Z, since they are no longer feasible possibilities for x.

We then get the following randomized algorithm for evaluating $f(x)$:

- Initialize $Z = f^{-1}(0)$.
- While $Z \neq \emptyset$:
  1. Calculate a from the entry-wise majority vote of Z.
  2. Determine $b \in \{0,1\}$ such that $\mathrm{R}(f_{a,b}) = O(\mathrm{Q}(f)^2)$ (this exists by Lemma 3.1).
  3. Run the randomized algorithm evaluating $f_{a,b}$ on x with some amplification (to be specified later).
  4. If its output is 1 (i.e. the algorithm thinks $x = a$ rather than $x \in f^{-1}(b)$), output $1-b$ and halt.
  5. If its output is 0, a bit i was queried to reveal $x_i \neq a_i$, so update Z (removing at least half its elements).
- If $Z = \emptyset$, output 1.

We note a few things about this algorithm. First, in step 3, notice that x need not be in the domain of $f_{a,b}$. However, we may still run the randomized algorithm that evaluates $f_{a,b}$, and use the fact that if x does happen to be in the domain (in particular, if $x \in f^{-1}(b)$), then the algorithm will work correctly. This is exactly what we use in step 4: if the algorithm that distinguishes a from $f^{-1}(b)$ says that x is equal to a, it need not mean that x is in fact equal to a, but it does mean that $x \notin f^{-1}(b)$.

Secondly, step 5 assumes that the randomized algorithm for evaluating $f_{a,b}$ will only conclude that an input x is not equal to a if it finds a disagreement with a. This is a safe assumption, as argued in Lemma 5 of [1].

Finally, we determine the number of queries this algorithm uses. The outer loop happens at most $\lfloor \log |f^{-1}(0)| \rfloor + 1 \leq \mathrm{Bal}(f)$ times. Step 3 in the loop is the only one which queries the input string. Since the loop repeats at most $\mathrm{Bal}(f)$ times, we can safely amplify the algorithm in step 3 $O(\log \mathrm{Bal}(f))$ times. This gives a query complexity of $O(Q(f)^2 \log \mathrm{Bal}(f))$ for step 3, so the overall number of queries is $O(Q(f)^2 \mathrm{Bal}(f) \log \mathrm{Bal}(f))$.

We can get rid of the log factor by being more careful with the amplification. Note that if we ever find a disagreement with a when running the algorithm, we may immediately stop amplifying and proceed to step 5. We keep a count c_0 of how many times we had to amplify in step 3 for functions of the form $f_{a,0}$, and a count c_1 for functions of the form $f_{a,1}$.

If c_0 ever reaches $2\,\mathrm{Bal}(f)$, we output 1 and halt. Similarly, if c_1 ever reaches $2\,\mathrm{Bal}(f)$, we output 0 and halt. This ensures the total amplification is $O(\mathrm{Bal}(f))$, so the total query complexity of the algorithm is $O(Q(f)^2 \mathrm{Bal}(f))$.

Note that if $f(x) = 0$ and the output of the algorithm was 1, it means that we ran the algorithm evaluating $f_{a,0}$ (for varying values of a) $2\,\mathrm{Bal}(f)$ times, and at most $\mathrm{Bal}(f)$ of those times the algorithm said that $x \in f^{-1}(0)$. For each individual run, the probability is at least $2/3$ that the algorithm would say that $x \in f^{-1}(0)$. An application of the Chernoff bound shows that the probability of this happening is exponentially small. Similarly, the probability of the algorithm giving 0 when in actuality $f(x) = 1$ is also exponentially small.

We conclude that $R(f) = O(Q(f)^2 \mathrm{Bal}(f))$, as desired. ◀

3.2 Application to Non-Sculptability

Theorem 3.2 immediately gives the following non-sculptability result, which says that unbalanced functions cannot be sculpted.

▶ **Corollary 3.3.** *Let $f : \{0,1\}^N \to \{0,1\}$ be a total function. For any promise $P \subseteq \{0,1\}^N$, we have*

$$\mathrm{R}(f|_P) = O(\mathrm{Q}(f|_P)^2 \mathrm{Bal}(f)).$$

Proof. Note that $\mathrm{Bal}(f|_P) \leq \mathrm{Bal}(f)$ for any f and P. Then, by Theorem 3.2, we have

$$\mathrm{R}(f|_P) = O(\mathrm{Q}(f|_P)^2 \mathrm{Bal}(f|_P)) = O(\mathrm{Q}(f|_P)^2 \mathrm{Bal}(f)).$$ ◀

We extend this result by showing that any function with a small number of large certificates also cannot be sculpted. This gives us a non-sculptability result in terms of the H-index of certificate complexity.

▶ **Theorem 3.4.** *Let $f : \{0,1\}^N \to \{0,1\}$ be a total function. For any promise $P \subseteq \{0,1\}^N$, we have*

$$\mathrm{R}(f|_P) = O(\mathrm{Q}(f|_P)^2 \mathrm{H}(\mathrm{C}^2_f)).$$

Proof. We design a deterministic algorithm that reduces the set of possibilities for the input to an unbalanced set. Specifically, the algorithm will reduce the possibilities for the input to a set $S \subseteq \{0,1\}^N$ such that $\mathrm{Bal}(f|_S) \leq \mathrm{H}(\mathrm{C}^2_f) + 1$. We then use Theorem 3.2 to get the desired non-sculptability result.

Note that every 1-certificate of f must conflict with every 0-certificate of f in at least one bit. Therefore, by querying all non-$*$ entries of a 0-certificate, we reveal at least one entry of each 1-certificate.

We design a deterministic algorithm for computing f on an input from P. The algorithm proceeds as follows: it repeatedly picks a 0-certificate p for f of size at most $\sqrt{\mathrm{H}(\mathrm{C}^2_f)}$ that is consistent with all the entries of the input that were revealed so far. It then queries all the non-$*$ entries of p. This is repeated $\sqrt{\mathrm{H}(\mathrm{C}^2_f)}$ times, or until there are no 0-certificates of size at most $\sqrt{\mathrm{H}(\mathrm{C}^2_f)}$ (whichever happens first). Finally, the algorithm returns the set S of strings that are consistent with the revealed entries of the input.

This algorithm uses at most $\mathrm{H}(\mathrm{C}^2_f)$ queries. We check its correctness by examining the set S. Clearly, the input is in S. Furthermore, if any certificate of f was revealed, then f is constant on S, so S contains either no 0-inputs or no 1-inputs.

There are at most $2^{\mathrm{H}(\mathrm{C}^2_f)}$ inputs with certificate complexity larger than $\sqrt{\mathrm{H}(\mathrm{C}^2_f)}$.

If the algorithm terminated because there were no consistent 0-certificates, then the only 0-inputs in S have certificates of size larger than $\sqrt{\mathrm{H}(\mathrm{C}^2_f)}$. There are at most $2^{\mathrm{H}(\mathrm{C}^2_f)}$ of them, so S has at most $2^{\mathrm{H}(\mathrm{C}^2_f)}$ 0-inputs to f. Conversely, if the algorithm went through $\sqrt{\mathrm{H}(\mathrm{C}^2_f)}$ iterations of querying consistent 0-certificates, then it must have revealed $\sqrt{\mathrm{H}(\mathrm{C}^2_f)}$ entries of each 1-certificate to f. If no 1-certificate was discovered, it means the revealed entries contradicted all 1-certificates of size at most $\sqrt{\mathrm{H}(\mathrm{C}^2_f)}$. Thus the only 1-inputs in S have certificate size greater than $\sqrt{\mathrm{H}(\mathrm{C}^2_f)}$, from which it follows that there are less than $2^{\mathrm{H}(\mathrm{C}^2_f)}$ of them.

We conclude that S contains either at most $2^{\mathrm{H}(\mathrm{C}^2_f)}$ 0-inputs to f or at most $2^{\mathrm{H}(\mathrm{C}^2_f)}$ 1-inputs to f. This gives $\mathrm{Bal}(f|_S) \leq \mathrm{H}(\mathrm{C}^2_f) + 1$.

We design a randomized algorithm for $f|_P$ as follows. First, we run the above deterministic algorithm to reduce the problem of computing $f|_P$ to the problem of computing $f|_{S\cap P}$. This costs $\mathrm{H}(\mathrm{C}^2_f)$ queries. By Theorem 3.2, there is a randomized algorithm that uses

$$O(\mathrm{Q}(f|_{S\cap P})^2 \,\mathrm{Bal}(f|_{S\cap P})) = O(\mathrm{Q}(f|_P)^2 \,\mathrm{Bal}(f|_S)) = O(\mathrm{Q}(f|_P)^2 \,\mathrm{H}(\mathrm{C}^2_f))$$

queries to compute $f|_{S\cap P}$. Running this algorithm allows us to compute $f|_P$. The total number of queries used was

$$O(\mathrm{Q}(f|_P)^2 \,\mathrm{H}(\mathrm{C}^2_f) + \mathrm{H}(\mathrm{C}^2_f)) = O(\mathrm{Q}(f|_P)^2 \,\mathrm{H}(\mathrm{C}^2_f)).$$ ◀

Note that Theorem 3.4 completes the first part of the proof of Theorem 1.2, since $\mathrm{H}(\mathrm{C}^2_f) \leq \mathrm{H}(\mathrm{C}_f)^2$. It is natural to wonder whether Theorem 3.4 is always at least as strong as Corollary 3.3. In Theorem 5.2, we will show that it is, up to a quadratic factor and a $\log N$ factor.

4 Sculpting from Communication Complexity

In this section, we show that if a function f has many inputs with large randomized certificate complexity then it *can* be sculpted: there is a promise P so that $f|_P$ exhibits a large quantum

speedup. This means that if $\mathrm{H}(\mathrm{RC}_f)$ is large, the function f can be sculpted. In Section 5, we will relate $\mathrm{H}(\mathrm{RC}_f)$ to $\mathrm{H}(\mathrm{C}_f)$, thereby completing the proof of Theorem 1.2.

Our sculptability proof will rely on the solution to a problem we call the "extended queries problem," which might be of independent interest. The solution to this problem will in turn use results from communication complexity.

4.1 The Extended Queries Problem

We usually let an algorithm for computing a (possibly partial) function $f : \{0,1\}^N \to \{0,1\}$ query the bits of the input x. But what happens if we let the algorithm make other types of queries? For example, if x is a Boolean string, we can let the algorithm query the parity of x. How does this extra power affect the query complexity of f? In particular, is there some special set of additional queries such that if a randomized algorithm is allowed to make the special queries, it can simulate *any* quantum algorithm? If so, how many special queries suffice for this property to hold?

To formalize this question, we need a few definitions.

▶ **Definition 4.1.** An *extension function with extension* G is an injective total function $\phi : \{0,1\}^N \to \{0,1\}^G$ (in particular, we need $G \geq N$).

An extension function specifies, for each input $x \in \{0,1\}^N$, the types of queries an algorithm is allowed to make on x. In other words, we will let algorithms query from $\phi(x)$ instead of from x. Note that the extension function may provide easy access to information about x that is hard to obtain otherwise (such as its parity).

▶ **Definition 4.2.** Let $f : \{0,1\}^N \to \{0,1\}$ be a partial function, and let ϕ be an extension function. The *extended version of* f *with respect to* ϕ is the partial function $f^\phi : \phi(\mathrm{Dom}(f)) \to \{0,1\}$ defined by $f^\phi(x) = f(\phi^{-1}(x))$.

Note that f^ϕ is a partial function from $\{0,1\}^G$ to $\{0,1\}$. We can consider $\mathrm{D}(f^\phi)$, $\mathrm{R}(f^\phi)$, $\mathrm{Q}(f^\phi)$, and so on. To pose the extended queries problem, we will need a notion of the complexity of a set of functions, defined as the maximum complexity of any function in that set.

▶ **Definition 4.3.** For any set of functions S, we define $\mathrm{D}(S) := \max_{f \in S} \mathrm{D}(f)$. We define $\mathrm{R}(S)$, $\mathrm{Q}(S)$, etc. similarly. Further, we define $\mathrm{D}^G(S)$, the *extended query complexity of* S *with extension* G, to be the minimum, over all extension functions ϕ with extension G, of $\max_{f \in S} \mathrm{D}(f^\phi)$. We define $\mathrm{R}^G(S)$, $\mathrm{Q}^G(S)$, etc. similarly.

In other words, for any set of functions, the extended query complexity of the set with G extension is the number of queries required to compute all functions in the set given the best possible extension. We observe that if $G \geq |S|$, the extended query complexity $\mathrm{D}^G(S)$ is 1, since the extension $\phi(x)$ for a given input x could simply specify the values of all the functions in S on x. We also observe that for all $G \geq N$, we have $\mathrm{D}^G(S) \leq \mathrm{D}(S)$, since the identity function is always a valid extension function. Moreover, the extended query complexity of a set is decreasing in G. We now ask the following question.

The Extended Queries Problem. Is there a set of functions S for which $\mathrm{Q}(S)$ is small but $\mathrm{R}^G(S)$ is large, even when the extension G is exponentially large in the input size N? We can also ask this question for other complexity measures, such as $\mathrm{R}(S)$ vs. $\mathrm{R}_0^G(S)$ or $\mathrm{R}_0(S)$ vs. $\mathrm{D}^G(S)$.

It turns out that a positive solution to the extended queries problem implies a sculptability result in terms of $\mathrm{H}(\mathrm{RC}_f)$, as the following theorem shows.

▶ **Theorem 4.4.** *Let $f : \{0,1\}^N \to \{0,1\}$ be a total function. Let $A = \frac{\mathrm{H}(\mathrm{RC}_f)}{4 \log N}$, and let S be any set of partial functions from $\{0,1\}^A$ to $\{0,1\}$. Then there is a promise $P \subseteq \{0,1\}^N$ such that*

$$\mathrm{Q}(f|_P) = O(\mathrm{Q}(S)), \qquad \mathrm{R}(f|_P) = \Omega(\mathrm{R}^N(S)).$$

Analogous statements hold for other pairs of complexity measures, such as D *and* R_0 *or* R_0 *and* R*.*

We delay the proof of Theorem 4.4 to Section 4.3. First, we settle the extended queries problem for R vs. Q: Theorem 4.6 will provide an exponential lower bound on G by reducing the extended queries problem to a problem in communication complexity.

4.2 Reducing Extension to Communication Complexity

For a partial function $f : \{0,1\}^{N_1} \times \{0,1\}^{N_2} \to \{0,1\}$, we will denote the communication complexities of f by $\mathrm{D}^{\mathrm{CC}}(f)$, $\mathrm{R}^{\mathrm{CC}}(f)$, $\mathrm{Q}^{\mathrm{CC}}(f)$, and $\mathrm{R}_0^{\mathrm{CC}}(f)$. We will use the following definition.

▶ **Definition 4.5.** Let $f : \{0,1\}^{N_1} \times \{0,1\}^{N_2} \to \{0,1\}$ be a partial function. For any $x \in \mathrm{Dom}(f)$, we write $x = x_A x_B$, with $x_A \in \{0,1\}^{N_1}$ and $x_B \in \{0,1\}^{N_2}$. Let $\mathrm{Dom}_A(f) = \{x_A : x \in \mathrm{Dom}(f)\}$ and $\mathrm{Dom}_B(f) = \{x_B : x \in \mathrm{Dom}(f)\}$. For any $a \in \mathrm{Dom}_A(f)$, we define the marginal of f with respect to a to be the partial function $f_a : \{0,1\}^{N_2} \to \{0,1\}$ defined by $f_a(b) := f(a,b)$ for all $b \in \{0,1\}^{N_2}$ such that $(a,b) \in \mathrm{Dom}(f)$. We define $\mathrm{Mar}(f) = \{f_a : a \in \mathrm{Dom}_A(f)\}$ to be the set of all marginal functions for f.

We now connect communication complexity to the extended queries problem.

▶ **Theorem 4.6.** *Let $f : \{0,1\}^{N_1} \times \{0,1\}^{N_2} \to \{0,1\}$ be a partial function. Then for all $G \geq N$,*

$$\mathrm{R}^G(\mathrm{Mar}(f)) = \Omega\left(\frac{\mathrm{R}^{\mathrm{CC}}(f)}{\log G}\right).$$

Similarly, we also have $\mathrm{D}^G(\mathrm{Mar}(f)) = \Omega(\mathrm{D}^{\mathrm{CC}}(f)/\log G)$, $\mathrm{R}_0^G(\mathrm{Mar}(f)) = \Omega(\mathrm{R}_0^{\mathrm{CC}}(f)/\log G)$, and $\mathrm{Q}^G(\mathrm{Mar}(f)) = \Omega(\mathrm{Q}^{\mathrm{CC}}(f)/\log G)$.

Proof. We prove the theorem for R. The statements for D, R_0, and Q will follow analogously. Let $\phi : \{0,1\}^{N_2} \to \{0,1\}^G$ be the best possible extension function, so that $\mathrm{R}^G(\mathrm{Mar}(f)) = \max_{g \in \mathrm{Mar}(f)} \mathrm{R}(g^\phi)$.

We now describe a randomized communication protocol for computing f. Alice receives a string a, and must compute $f(a,b)$, where b is Bob's string. This is equivalent to computing $f_a(b)$. Since Alice knows f_a, she also knows f_a^ϕ. Let R a randomized algorithm that computes f_a^ϕ using at most $\mathrm{R}^G(\mathrm{Mar}(f))$ queries. Alice will run this algorithm, and for each query, she will send the index of that query to Bob (as a number between 1 and G). Bob will reply with the corresponding bit of $\phi(y)$ (as a bit in {0,1}). This allows Alice to compute $f_a(b) = f(a,b)$.

The total communication in this protocol is at most $(\lceil \log G \rceil + 1)\,\mathrm{R}^G(\mathrm{Mar}(f))$. Since this upper-bounds the randomized communication complexity of f (using private coins), the desired result follows. ◀

Theorem 4.6 allows us to use communication complexity as a tool for lower-bounding the extended query complexity of certain sets of functions. To use it to solve the extended queries problem, we need a function f that has large randomized communication complexity but for which $\mathrm{Q}(\mathrm{Mar}(f))$ is small. To construct such a function, we start from a simple function that was recently shown to separate randomized from quantum communication complexity, called the Vector in Subspace problem.

The Vector in Subspace Problem. In this problem, Bob gets a unit vector $v \in \mathbb{R}^n$, and Alice gets a subspace H of $\mathbb{R}^n$ of dimension $n/2$. It is promised that either $v \in H$ or $v \in H^\perp$; the task is to determine which is the case. We assume for simplicity that n is a power of 2.

This problem was first studied in [12] and was also described in [15]. Klartag and Regev [11] showed that this problem has randomized communication complexity $\Omega(n^{1/3})$. In addition, it is easy to see that the one-way quantum communication complexity of the problem is at most $\log n$: Bob can send a superposition over $\log n$ bits with amplitudes determined by v; Alice can then apply the projective measurement given by $(H, H^\perp)$.

To apply this function to the extended queries problem, we need a few modifications. First, we need a discrete version of the problem. [11] showed that a lower bound of $\Omega(n^{1/3})$ for randomized communication complexity applies to a discrete version of the problem in which each real number is described using $O(\log n)$ bits; that is, Alice's subspace is given using $n/2$ vectors of length n, whose entries are specified using $O(\log n)$ bits, and Bob's vector is specified using n real numbers of $O(\log n)$ bits each.

$\mathrm{Mar}(f)$ is the set of functions where we know Alice's subspace H, and are allowed to query from Bob's input vector. However, phrased this way, it is not clear how to use a quantum algorithm to compute such functions using few queries. To solve this problem, we modify the way Bob's input is specified. Instead of specifying only the entries to the vector, Bob's input string also lists some "partial sums" of the vector entries.

The idea is for Bob's vector to allow Alice to use the following algorithm to construct the state with amplitudes specified by v. We interpret v as specifying a superposition over strings of length $\log n$. Alice starts by querying the probability p that the first bit of this string is 0 when this state is measured. Alice will now place a $\sqrt{p}$ amplitude on querying the probability that the second bit is 0 conditioned on the first bit being 0, and a $\sqrt{1-p}$ amplitude on querying the probability that the second bit is 0 conditioned on the first bit being 1. Alice keeps going in this way, until she gets to the final bit of the string of length $\log n$, at which point she queries the phase. This allows her to construct the state determined by the amplitudes in v.

Of course, for this to work, Bob's input must provide all of these conditional probabilities. There is one such probability to specify for the first bit, two for the second, four for the third, and so on. Since there are $\log n$ bits, Bob's input needs to specify only $O(n)$ probabilities. Each can be specified with $O(\log n)$ precision, so Bob's total input size is $O(n \log n)$. Moreover, Alice constructs the desired state after $O(\log n)$ queries to the probabilities, or $O(\log^2 n)$ queries to the bits of Bob's input.

We thus get the following theorem.

▶ **Theorem 4.7.** *For all $A \in \mathbb{N}$, there is a set S of partial functions from $\{0,1\}^A$ to $\{0,1\}$ such that for all $G \geq A$,*

$$\mathrm{Q}(S) = O(\log^2 A), \quad \mathrm{R}^G(S) = \Omega\left(\frac{A^{1/3}}{\log^{1/3} A \cdot \log G}\right).$$

Proof. Let f be the function described above with $n = A/\log A$, and let $S = \mathrm{Mar}(f)$. Then $\mathrm{Q}(S) = O(\log^2 n) = O(\log^2 A)$ and $\mathrm{R}^{\mathrm{CC}}(f) = \Omega(n^{1/3}) = \Omega(A^{1/3}/\log^{1/3} A)$. By Theorem 4.6, we get $\mathrm{R}^G(S) = \Omega(A^{1/3}/(\log^{1/3} A \cdot \log G))$. ◀

Together with Theorem 4.4, this implies that any function with large $\mathrm{H}(\mathrm{RC}_f)$ can be sculpted, simply by plugging S from Theorem 4.7 into Theorem 4.4 and setting $G = N$.

4.3 Reducing Sculpting to Extended Query Complexity

We now prove Theorem 4.4, restated here for convenience.

▶ **Theorem 4.4.** *Let $f : \{0,1\}^N \to \{0,1\}$ be a total function. Let $A = \frac{\mathrm{H}(\mathrm{RC}_f)}{4 \log N}$, and let S be any set of partial functions from $\{0,1\}^A$ to $\{0,1\}$. Then there is a promise $P \subseteq \{0,1\}^N$ such that*

$$\mathrm{Q}(f|_P) = O(\mathrm{Q}(S)), \qquad \mathrm{R}(f|_P) = \Omega(\mathrm{R}^N(S)).$$

Analogous statements hold for other pairs of complexity measures, such as D *and* R_0 *or* R_0 *and* R*.*

Proof. There are at least $2^{\mathrm{H}(\mathrm{RC}_f \cdot 2 \log N)}$ inputs x with $\mathrm{RC}_f(x) \geq \mathrm{H}(\mathrm{RC}_f \cdot 2 \log N)/(2 \log N)$. Let the set of such inputs be C. By Lemma 2.1, if $N \geq 2$, there is a set B of

$$\frac{\mathrm{H}(\mathrm{RC}_f \cdot 2 \log N)}{2 \log N} \geq \frac{\mathrm{H}(\mathrm{RC}_f)}{2 \log N}$$

indices in $\{1, 2, \ldots, N\}$ that is shattered by the inputs in C. We'll restrict B to have size at most $\mathrm{H}(\mathrm{RC}_f)/(4 \log N)$, so $|B| = A$. Let $\phi : \{0,1\}^A \to \{0,1\}^N$ be defined by mapping each string $x \in \{0,1\}^A$ to a string z in C such that restricting z to A gives x. This is an injective mapping, so ϕ is an extension function with extension size N.

Next, consider the set S of partial Boolean functions from $\{0,1\}^A$ to $\{0,1\}$. Let $S^\phi = \{g^\phi : g \in S\}$. Then $\mathrm{R}(S^\phi) \geq \mathrm{R}^N(S)$. It follows that there is some function $g^\phi \in S^\phi$ such that $\mathrm{R}(g^\phi) \geq \mathrm{R}^N(S)$.

We will use the function g^ϕ to define the desired promise P. The domain of g^ϕ is contained in C. Let x be in this domain, so that $\mathrm{RC}_f(x) \geq \mathrm{H}(\mathrm{RC}_f \cdot 2 \log N)/(2 \log N) \geq 2A$. Let μ_x be a distribution over inputs y such that $f(x) \neq f(y)$, with the property that for any bit i, $\Pr[y_i \neq x_i] \leq 1/\mathrm{RC}_f(x) \leq 1/(2A)$. Then for all $x \in C$, a randomized algorithm has a hard time distinguishing between x and μ_x. For each such x, let μ'_x be the distribution μ_x conditioned on the sampled input agreeing with x on the bits in B. Since the probability of an input sampled from μ_x disagreeing with x on B is at most $|B| \cdot 1/(2A) \leq 1/2$, the distribution μ'_x is not too far from μ_x. In particular, any randomized algorithm that finds a disagreement with x on an input sampled from μ'_x with probability p will also find a disagreement with x on an input sampled from μ_x with probability at least $p/2$. It follows that a randomized algorithm must use $\Omega(A)$ queries to distinguish x from μ'_x.

We construct the promise P as follows. Start with $P = \emptyset$. For each $x \in \mathrm{Dom}(g^\phi)$, we add x to P if $f(x) = g^\phi(x)$; otherwise, we add the support of μ'_x to P.

It remains to lower-bound $\mathrm{R}(f|_P)$ and to upper-bound $\mathrm{Q}(f|_P)$. We start with the upper bound. Let $y \in P$, and consider the value of y on B. If x is an input of the domain of g^ϕ that caused y to be added, then x and y agree on B. Further, the values of x on B are simply $\phi^{-1}(x) \in \{0,1\}^{|B|}$, and $g(\phi^{-1}(x)) = g^\phi(x) = f(y)$. This means $g(y|_B) = f(y)$. We now have the quantum algorithm work only with the bits of $y|_B$, ignoring the rest. The

algorithm need only compute $g(y|_B)$. Since $g \in S$, we get $\mathrm{Q}(f|_P) \leq \mathrm{Q}(g) \leq \mathrm{Q}(S)$, as desired. A similar argument would upper-bound other complexity measures, such as R, R_0, or D.

For the lower bound, consider the hard distribution μ on inputs to g^ϕ obtained from Yao's minimax principle [19]. This distribution has the property that any randomized algorithm for g^ϕ that succeeds with probability at least $2/3$ on inputs sampled from μ must use $\mathrm{R}(g^\phi)$ queries. We construct a new distribution μ' over P by generating an element $x \in \mathrm{Dom}(g^\phi)$ according to μ, and then outputting either x or a sample from μ'_x, depending on which of them was added to P. We lower-bound the number of queries a randomized algorithm requires to compute f on an input sampled from μ' by a reduction from either computing g^ϕ on inputs sampled from μ, or else distinguishing x from μ'_x.

Let R be a randomized algorithm for $f|_P$. Let $x \sim \mu$. We wish to compute $g^\phi(x)$. Although x may not be in P, consider running R on x anyway. The algorithm will correctly output $g^\phi(x)$ with some probability p, depending on both the internal randomness of R and on μ. If $p \geq 3/5$, we could amplify R a constant number of times to turn it into an algorithm for g that works on inputs sampled from the hard distribution μ, which means R must use $\Omega(\mathrm{R}(g^\phi)) = \Omega(\mathrm{R}^N(S))$ queries. So suppose that $p \leq 3/5$.

Next, given $x \sim \mu$, we let y_x be either x or a sample from μ'_x, as μ' dictates. Then running R on y_x gives $f(y_x) = g^\phi(x)$ with probability at least $2/3$. On the other hand, running R on x gives output $g^\phi(x)$ with probability at most $3/5$. That is, we have

$$\Pr_{R,x\sim\mu}[R(x) = g^\phi(x)] = \mathop{\mathbb{E}}_{x\sim\mu}\left[\Pr_R[R(x) = g^\phi(x)]\right] \leq 3/5$$

$$\Pr_{R,x\sim\mu,y_x}[R(y_x) = g^\phi(x)] = \mathop{\mathbb{E}}_{x\sim\mu}\left[\Pr_{R,y_x}[R(y_x) = g^\phi(x)]\right] \geq 2/3$$

From which it follows that

$$\mathop{\mathbb{E}}_{x\sim\mu}\left[\Pr_{R,y_x}[R(y_x) = g^\phi(x)] - \Pr_R[R(x) = g^\phi(x)]\right] \geq 1/15.$$

This means there must be some specific input $\hat{x}$ such that the probability of R outputting $g^\phi(\hat{x})$ when run on $y_{\hat{x}}$ is at least $1/15$ more than the probability of R outputting $g^\phi(\hat{x})$ when run on $\hat{x}$. In particular, we must have $y_{\hat{x}} \neq \hat{x}$, so $y_{\hat{x}}$ is a sample from $\mu'_{\hat{x}}$. Therefore, R distinguishes $\hat{x}$ from $\mu'_{\hat{x}}$ with constant probability, so it uses at least $\Omega(A)$ queries.

We conclude that $\mathrm{R}(f|_P) = \Omega(\min\{\mathrm{R}^N(S), A\})$. Since the domain of the functions in S is $\{0,1\}^A$, their query complexity is at most A. Thus $\mathrm{R}(f|_P) = \Omega(\mathrm{R}^N(S))$, as desired. A similar argument lower-bounds other complexity measures, such as R_0 or D. ◀

This proof uses the fact that RC lower-bounds R, so it would not work on complexity measures that are not lower-bounded by RC (for example, $\mathrm{C}^{(1)}$). For Q, it might be possible to use a similar argument and suffer a quadratic loss, since Q is lower-bounded by $\sqrt{\mathrm{RC}}$. However, since there is no hard distribution for a quantum query complexity problem, this might be trickier to prove (we will not need it in this paper).

We can use the previous theorems to get a sculptability result for R vs. Q in terms of the H-index of randomized certificate complexity.

▶ **Corollary 4.8.** *Let $f : \{0,1\}^N \to \{0,1\}$ be a total function. Then there is a promise $P \subseteq \{0,1\}^N$ such that*

$$\mathrm{Q}(f|_P) = O(\log^2 \mathrm{H}(\mathrm{RC}_f)), \qquad \mathrm{R}(f|_P) = \Omega\left(\frac{\mathrm{H}(\mathrm{RC}_f)^{1/3}}{\log^{5/3} N}\right).$$

Proof. This follows from Theorem 4.4 together with Theorem 4.7. ◀

To complete the proof of Theorem 1.2, all that remains is relating $\mathrm{H}(\mathrm{RC}_f)$ to $\mathrm{H}(\mathrm{C}_f)$.

5 Relating $\mathrm{H}(\mathrm{C}_f)$, $\mathrm{H}(\mathrm{RC}_f)$, and $\mathrm{H}(\mathrm{bs}_f)$

In this section, we relate $\mathrm{H}(\mathrm{C}_f)$ to $\mathrm{H}(\mathrm{RC}_f)$, completing the characterization of sculpting. Actually, we will prove a relationship between $\mathrm{H}(\mathrm{C}_f)$ and $\mathrm{H}(\mathrm{bs}_f)$, which implies the desired relationship since $\mathrm{H}(\mathrm{bs}_f) \leq \mathrm{H}(\mathrm{RC}_f)$. The proof is somewhat involved, but splits naturally into three parts. In Lemma 5.1, we show a relationship between $\mathrm{C}_f(x)$ and $\mathrm{RC}_f(x)$ in terms of the number of 0- and 1-inputs of f. In Theorem 5.2, we show that $\mathrm{H}(\mathrm{C}_f) = O(\mathrm{Bal}(f) \log N)$. Finally, Theorem 5.3 gives the desired relationship between $\mathrm{H}(\mathrm{C}_f)$ and $\mathrm{H}(\mathrm{bs}_f)$.

▶ **Lemma 5.1.** *Let $f : \{0,1\}^N \to \{0,1\}$ be a partial function, and let $x \in \mathrm{Dom}(f)$. If $f(x) = 0$, then*

$$\mathrm{C}_f(x) \leq \mathrm{RC}_f(x)(1 + \log |f^{-1}(1)|)$$

and if $f(x) = 1$, then

$$\mathrm{C}_f(x) \leq \mathrm{RC}_f(x)(1 + \log |f^{-1}(0)|).$$

Proof. For $x \in f^{-1}(1)$, we wish to upper-bound $\mathrm{C}_f(x)$ in terms of $\mathrm{RC}_f(x)$, assuming $f^{-1}(0)$ is small. A certificate for x consists of a partial assignment of x that contradicts all the elements of $f^{-1}(0)$.

Consider the greedy strategy for certifying x, which works by repeatedly choosing the bit of x that contradicts as many of the 0-inputs as possible, and adding it to the certificate. By definition, this strategy produces a certificate for x of size at least $\mathrm{C}_f(x)$.

Let p_i be the fraction of the remaining inputs which are contradicted by the i-th bit of the greedy algorithm. The number of remaining inputs during the run of the greedy algorithm is then

$$|f^{-1}(0)|,\ |f^{-1}(0)|(1-p_1),\ |f^{-1}(0)|(1-p_1)(1-p_2), \ldots$$

The number of remaining inputs in the greedy algorithm will be upper-bounded by a geometric sequence that starts at $|f^{-1}(0)|$ and has ratio $1 - \min_i p_i$. Such a sequence decreases to 1 after at most

$$\frac{-1}{\log(1 - \min_i p_i))}(1 + \log |f^{-1}(0)|) \leq \frac{1 + \log |f^{-1}(0)|}{\min_i p_i}$$

steps. It follows that

$$\mathrm{C}_f(x) \leq \frac{1 + \log |f^{-1}(0)|}{\min_i p_i}.$$

It remains to show that $\mathrm{RC}_f(x) = \Omega(1/\min_i p_i)$. Let j be the step of the greedy algorithm that achieves this minimum, i.e. $p_j = \min_i p_i$. Then before the j^{th} step of the algorithm, there is a non-empty set S of 0-inputs for f such that for any bit of the input, at most a p_j fraction of the elements of S disagree with x on that bit. In other words, x is entry-wise very close to the "average" of the elements of S. If we give each element of S weight $1/(p_j|S|)$, we would get a feasible set of fractional blocks with total weight $1/p_j$. Thus $\mathrm{RC}_f(x) \geq 1/p_j$, so $\mathrm{C}_f(x) \leq \mathrm{RC}_f(x)(\log |f^{-1}(0)| + 1)$. An analogous argument works when x is a 0-input to f. ◀

▶ **Theorem 5.2.** *Let $N \geq 2$, and let $f : \{0,1\}^N \to \{0,1\}$ be a total function. Then*

$$\mathrm{H}(\mathrm{C}_f) \leq 10\,\mathrm{Bal}(f) \log N.$$

Proof. Without loss of generality, suppose $|f^{-1}(0)| \leq |f^{-1}(1)|$. The number of 0-inputs with large certificates is at most $|f^{-1}(0)| \leq 2^{\mathrm{Bal}(f)}$. Let S be the set of 1-inputs with certificates of size greater than $5\,\mathrm{Bal}(f)$. We wish to show that S is small. Lemma 2.1 implies there is a set $B = \{i_1, i_2, \ldots, i_{|B|}\}$ of indices of the input of size at least $\log|S|/(2\log N)$ that is shattered by S. Therefore, to show that S is small, it suffices to show that B is small.

From Lemma 5.1, we have $\mathrm{C}_f(x) \leq \mathrm{RC}_f(x)\,\mathrm{Bal}(f)$ for any 1-input x, so for all $x \in S$, we have $\mathrm{RC}_f(x) \geq \mathrm{C}_f(x)/\,\mathrm{Bal}(f) > 5$. This means for all $x \in S$, there is a distribution μ_x over 0-inputs such that for each i, the probability that $y_i \neq x_i$ when y is sampled from μ_x is less than $1/5$.

Let μ_B be the uniform distribution over B. Let $\delta(b,c) = 1$ if $b \neq c$ and 0 otherwise. We then write

$$\frac{1}{5} > \mathop{\mathbb{E}}_{i\sim\mu_B}\left(\mathop{\mathbb{E}}_{y\sim\mu_x}\delta(x_i,y_i)\right) = \mathop{\mathbb{E}}_{y\sim\mu_x}\left(\mathop{\mathbb{E}}_{i\sim\mu_B}\delta(x_i,y_i)\right).$$

We can conclude that for any $x \in S$, there exists a 0-input y_x that differs from x in less than one fifth of the bits of B. In other words, the distance between $x|_B$ and $y_x|_B$ is less than $|B|/5$. The idea is now to upper-bound $|B|$ by using the fact that for every string in $\{0,1\}^{|B|}$ there is a 0 input y such that $y|_B$ is close to that string, and there are few 0-inputs overall. Indeed, the number of strings in $\{0,1\}^{|B|}$ is $2^{|B|}$, and each 0-input can only be of distance less than $|B|/5$ from $2^{H(1/5)|B|}$ of them (where $H(1/5)$ denotes the entropy of $1/5$). Therefore, to cover all the strings in $\{0,1\}^{|B|}$, there must be more than $2^{(1-H(1/5))|B|}$ 0-inputs. Then

$$\mathrm{Bal}(f) \geq \log|f^{-1}(0)| > (1-H(1/5))|B| \geq (1-H(1/5))\frac{\log|S|}{2\log N},$$

so

$$\log|S| < \frac{2\,\mathrm{Bal}(f)\log N}{1-H(1/5)} \leq 8\,\mathrm{Bal}(f)\log N.$$

This means there are less than $2^{8\,\mathrm{Bal}(f)\log N}$ 1-inputs with certificate size at least $5\,\mathrm{Bal}(f)$. There are also at most $2^{\mathrm{Bal}(f)}$ 0-inputs with certificate size at least $5\,\mathrm{Bal}(f)$ (because there are at most that many 0-inputs in total). Thus the log of the total number of inputs with certificates larger than $5\,\mathrm{Bal}(f)$ is at most $10\,\mathrm{Bal}(f)\log N$. It follows that $\mathrm{H}(\mathrm{C}_f) \leq 10\,\mathrm{Bal}(f)\log N$. ◀

▶ **Theorem 5.3.** *Let $f : \{0,1\}^N \to \{0,1\}$ be a total function. Then*

$$\mathrm{H}(\mathrm{C}_f) = O(\mathrm{H}(\mathrm{bs}_f)^2 \log N).$$

Proof. Let A be the set of inputs that have certificate size more than $\mathrm{H}(\mathrm{C}_f)$. Let A_0 be the set of 0-inputs in A, and let A_1 be the set of 1-inputs in A. Let B be the set of inputs that have block sensitivity more than b, with $b = \sqrt{\mathrm{H}(\mathrm{C}_f)/2}$. Let B_0 be the set of 0-inputs in B, and let B_1 be the set of 1-inputs in B. Without loss of generality, assume $|A_0| \geq |A_1|$. Since $|A| \geq 2^{\mathrm{H}(\mathrm{C}_f)}$, we have $|A_0| \geq 2^{\mathrm{H}(\mathrm{C}_f)-1}$.

Now, let $g : \{0,1\}^N \to \{0,1\}$ be the total function defined by $g(x) = 1$ if and only if $x \in B_1$. Suppose x is an element of $A_0 \backslash B_0$. Consider certifying that x is a 0-input for g; let p be the smallest such certificate. Then p is consistent with x but inconsistent with all the strings in B_1. We claim that this certificate must be large: its size must be greater than $\mathrm{H}(\mathrm{C}_f) - b^2 = \mathrm{H}(\mathrm{C}_f)/2$. To show this, we show that we can turn p into a certificate for x with respect to f (instead of with respect to g) by adding only b^2 bits to it.

Let q be a minimal sensitive block of x (with respect to f) that is disjoint from p. Since x is a 0-input for f, x^q is a 1-input for f. Since q is disjoint from p, x^q is consistent with p, so $x^q \notin B_1$. Thus the block sensitivity of x^q is at most b. However, since q was a minimal sensitive block, the sensitivity of x^q is at least $|q|$; thus $|q| \leq b$. It follows that all minimal sensitive blocks of x that are disjoint from p must have size at most b.

In addition, since $x \in A_0 \backslash B_0$, the block sensitivity of x is at most b. We can now construct a certificate for x by taking a maximal set of minimal disjoint sensitive blocks for x, all of which are disjoint from p. There will be at most b such blocks, and each will have size at most b. Therefore, this certificate for x has size at most $|p| + b^2$. Since $x \in A_0$, we must have $|p| + b^2 > \mathrm{H}(\mathrm{C}_f)$, or $|p| > \mathrm{H}(\mathrm{C}_f) - b^2 = \mathrm{H}(\mathrm{C}_f)/2$. We have shown that the elements of $A_0 \backslash B_0$ all have certificate size greater than $\mathrm{H}(\mathrm{C}_f)/2$ even with respect to g.

Now, by Theorem 5.2, the number of inputs x that have certificate size more than $10(1 + \log |B_1|) \log N$ with respect to g is at most $2^{10(1+\log |B_1|) \log N}$. It follows that either $\mathrm{H}(\mathrm{C}_f)/2 \leq 10(1 + \log |B_1|) \log N$ (so that the theorem doesn't apply), or else $|A_0 \backslash B_0| \leq 2^{10(1+\log |B_1|) \log N}$.

In the former case, we have

$$\log |B| \geq \log |B_1| \geq \frac{\mathrm{H}(\mathrm{C}_f)}{20 \log N} - 1.$$

In the latter case, we have

$$2^{\mathrm{H}(\mathrm{C}_f)-1} \leq |A_0| \leq |B_0| + 2^{10(1+\log |B_1|) \log N} = |B_0| + (2|B_1|)^{10 \log N} \leq (2|B|)^{10 \log N},$$

so in that case,

$$\log |B| \geq \frac{\mathrm{H}(\mathrm{C}_f) - 1}{10 \log N} - 1.$$

Thus, in both cases,

$$\log |B| \geq \frac{\mathrm{H}(\mathrm{C}_f) - 1}{20 \log N} - 1 = \Omega\left(\frac{\mathrm{H}(\mathrm{C}_f)}{\log N}\right).$$

This means there are $2^{\Omega(\mathrm{H}(\mathrm{C}_f)/\log N)}$ inputs with block sensitivity more than $\sqrt{\mathrm{H}(\mathrm{C}_f)/2}$. We thus have

$$\mathrm{H}(\mathrm{bs}_f) = \Omega\left(\min\left\{\frac{\mathrm{H}(\mathrm{C}_f)}{\log N}, \sqrt{\mathrm{H}(\mathrm{C}_f)}\right\}\right) = \Omega\left(\sqrt{\frac{\mathrm{H}(\mathrm{C}_f)}{\log N}}\right). \quad \blacktriangleleft$$

Theorem 1.2 now follows from Theorem 3.2 (the non-sculptability theorem in terms of $\mathrm{H}(\mathrm{C}_f^2)$), Corollary 4.8 (the sculptability result in terms of $\mathrm{H}(\mathrm{RC}_f)$), and Theorem 5.3 (relating $\mathrm{H}(\mathrm{bs}_f)$ to $\mathrm{H}(\mathrm{C}_f)$), together with the properties that $\mathrm{H}(\mathrm{C}_f^2) \leq \mathrm{H}(\mathrm{C}_f)^2$ and that $\mathrm{H}(\mathrm{bs}_f) \leq \mathrm{H}(\mathrm{RC}_f)$. We restate Theorem 1.2 here for convenience.

▶ **Theorem 1.2.** *For all total functions $f : \{0,1\}^N \to \{0,1\}$ and all promises $P \subseteq \{0,1\}^N$, we have*

$$\mathrm{R}(f|_P) = O(\mathrm{Q}(f|_P)^2 \, \mathrm{H}(\mathrm{C}_f)^2).$$

Conversely, for all total functions $f : \{0,1\}^N \to \{0,1\}$, there is a promise $P \subseteq \{0,1\}^N$ such that

$$\mathrm{R}(f|_P) = \Omega\left(\frac{\mathrm{H}(\mathrm{C}_f)^{1/6}}{\log^{11/6} N}\right) \qquad \textit{and} \qquad \mathrm{Q}(f|_P) = O(\log^2 \mathrm{H}(\mathrm{C}_f)).$$

Theorem 1.1 follows as a corollary. This completes the proof of our main result.

6 Sculpting Randomized Speedups

Now that we've characterized sculpting quantum query complexity, we turn our attention to sculpting other measures. Recall that

$$\mathrm{Q}(f) \leq \mathrm{R}(f) \leq \mathrm{R}_0(f) \leq \mathrm{D}(f).$$

We showed that sculpting $\mathrm{R}(f)$ vs. $\mathrm{Q}(f)$ is possible if and only if f has a large number of large certificates. We now show that the exact same condition characterizes sculpting $\mathrm{D}(f)$ vs. $\mathrm{R}_0(f)$. On the other hand, we show that $\mathrm{R}_0(f)$ vs. $\mathrm{R}(f)$ behaves differently: it's *always* possible to sculpt a function f to a promise P such that $\mathrm{R}(f|_P)$ is constant and $\mathrm{R}_0(f|_P)$ is almost as large as $\mathrm{R}_0(f)$.

We start by characterizing sculpting for D vs. R_0.

6.1 Sculpting D vs. R_0

The proof of this characterization will follow that of Theorem 1.2. For the non-sculptability direction, we need an analogue of Theorem 3.2, relating deterministic and zero-error randomized query complexities in terms of $\mathrm{Bal}(f)$. We prove the following theorem.

▶ **Theorem 6.1.** *Let $f : \{0,1\}^N \to \{0,1\}$ be a partial function. Then*

$$D(f) \leq 2R_0(f)\,\mathrm{Bal}(f).$$

Proof. Consider the zero-error randomized algorithm that takes $\mathrm{R}_0(f)$ expected queries to evaluate f. By Markov's inequality, if we let this algorithm make $2\,\mathrm{R}_0(f)$ queries on input x, it will succeed in computing $f(x)$ (and provide a certificate for x) with probability at least $1/2$. This gives us a probability distribution μ over deterministic algorithms, each of which makes $2\,\mathrm{R}_0(f)$ queries, such that for each input x the probability that an algorithm sampled from μ finds a certificate when run on x is at least $1/2$.

For a deterministic algorithm D and an input x, let $c(D,x) = 1$ if D finds a certificate for x, and $c(D,x) = 0$ otherwise. Let $Z \subseteq \{0,1\}^N$. Then

$$\mathop{\mathbb{E}}_{D\sim\mu}\left[\sum_{x\in Z} c(D,x)\right] = \sum_{x\in Z} \mathop{\mathbb{E}}_{D\sim\mu}[c(D,x)] \geq \sum_{x\in Z}(1/2) = \frac{|Z|}{2}.$$

It follows that there is a deterministic algorithm D_Z that makes $2\,\mathrm{R}_0(f)$ queries and finds a certificate when run on half the inputs in Z.

Suppose without loss of generality that $|f^{-1}(0)| \leq |f^{-1}(1)|$. Now, on input x, set $Z = f^{-1}(0)$, and run D_Z on x. If it fails to find a certificate, then we have eliminated half of Z as possibilities for the input. Repeating this $\lfloor \log |f^{-1}(0)| \rfloor + 1 \leq \mathrm{Bal}(f)$ times suffices to eliminate all of $f^{-1}(0)$ as possibilities for x, and hence to determine the value of $f(x)$. The total number of queries used is at most $2\,\mathrm{R}_0(f)\,\mathrm{Bal}(f)$. ◀

Note that Theorem 6.1 and Theorem 3.2 together complete the proof of Theorem 1.4.

Next, we turn Theorem 6.1 into a non-sculptability theorem in terms of $\mathrm{H}(\mathrm{C}_f)$. The argument in Theorem 3.4 follows verbatim, and we get the following sculpting lower bound.

▶ **Corollary 6.2.** *Let $f : \{0,1\}^N \to \{0,1\}$ be a total function. For any promise $P \subseteq \{0,1\}^N$, we have*

$$\mathrm{D}(f|_P) = O(\mathrm{R}_0(f|_P)\,\mathrm{H}(\mathrm{C}_f)^2).$$

We now prove the other direction: we show that sculpting is possible when $\mathrm{H}(\mathrm{RC}_f)$ is large. Using the arguments from Section 4, it suffices to solve the extended queries problem for D vs. R_0. We do this using the reduction to communication complexity in Theorem 4.6.

▶ **Theorem 6.3.** *For all $N \in \mathbb{N}$, there is a set of partial functions S from $\{0,1\}^N$ to $\{0,1\}$ such that for all $G \geq N$,*

$$\mathrm{R}_0(S) = O(1), \qquad \mathrm{D}^G(S) = \Omega\left(\frac{N}{\log G}\right).$$

Proof. We start with Equality, in which Alice and Bob are each given an n-bit string and wish to know if their strings are equal. This problem has deterministic query complexity $\Omega(n)$, but small randomized query complexity. To make the zero-error randomized query complexity small as well, we give Alice and Bob two strings each, with the promise that either their first strings are equal and the second strings are not, or vice versa. The goal will be to determine which is the case. It is not hard to see that the deterministic communication complexity of this problem is still $\Omega(n)$.

We need to get the zero-error randomized query complexity of the marginal functions to be small. To do this, we introduce another modification: we encode each of Bob's strings using a fixed random code of length $3n$. This code will have the property that the distance between any pair of codewords is $\Omega(n)$. To compute a marginal function f_{a_1,a_2} indexed by Alice's strings, we can simply randomly sample from each of Bob's strings; after $O(1)$ samples, we will discover which of his strings do not match the codeword corresponding to a_1 and a_2.

This construction gives us a function $f : \{0,1\}^{2n} \times \{0,1\}^{6n} \to \{0,1\}$ such that $\mathrm{D}^{\mathrm{CC}}(f) = \Omega(n)$ and $\mathrm{R}_0(\mathrm{Mar}(f)) = O(1)$. Setting $N = 6n$ and using Theorem 4.6 finishes the proof. ◀

Putting this together, we get the following sculpting theorem which, together with Corollary 6.2, is analogous to Theorem 1.2.

▶ **Theorem 6.4.** *For all total functions $f : \{0,1\}^N \to \{0,1\}$, there is a promise $P \subseteq \{0,1\}^N$ such that*

$$\mathrm{D}(f|_P) = \Omega\left(\frac{\sqrt{\mathrm{H}(\mathrm{C}_f)}}{\log^{5/2} N}\right) \qquad \text{and} \qquad \mathrm{R}_0(f|_P) = O(1).$$

Proof. This follows from Theorem 6.3 together with Theorem 4.4 and Theorem 5.3. ◀

We also get the following corollary, analogous to Theorem 1.1.

▶ **Corollary 6.5.** *Let $f : \{0,1\}^N \to \{0,1\}$ be a total function. Then there is a promise $P \subseteq \{0,1\}^N$ such that $\mathrm{D}(f|_P) = N^{\Omega(1)}$ and $\mathrm{R}_0(f|_P) = N^{o(1)}$, if and only if $\mathrm{H}(\mathrm{C}_f) = N^{\Omega(1)}$. Futhermore, in this case we also have $\mathrm{R}_0(f|_P) = O(1)$.*

6.2 Sculpting R_0 vs. R

While it is possible to use the above argument to get a sculptability result for R_0 vs. R, we can get a stronger result by a direct argument. In fact, unlike R vs. Q or D vs. R_0, sculpting R_0 vs. R is *always* possible (there is no dependence on any H-index).

▶ **Theorem 6.6.** *Let $f : \{0,1\}^N \to \{0,1\}$ be a non-constant total function. Then there is a promise $P \subseteq \{0,1\}^N$ such that*

$$\mathrm{R}(f|_P) = 1, \qquad \mathrm{R}_0(f|_P) \geq \frac{\mathrm{R}_0(f)^{1/3}}{6}.$$

Proof. We actually prove a stronger result, finding a promise P such that $\mathrm{R}(f|_P) = 1$ and $\mathrm{R}_0(f|_P) \geq \mathrm{bs}(f)/6$. We then use the known relationship $\mathrm{R}_0(f) \leq \mathrm{bs}(f)^3$ for total functions to get the desired result. Note that finding P with $\mathrm{R}(f|_P) = 1$ and $\mathrm{R}_0(f|_P) \geq \mathrm{bs}(f)/6$ is trivial when $\mathrm{bs}(f) \leq 6$; thus we assume $\mathrm{bs}(f) > 6$.

Let $x \in \{0,1\}^N$ be such that $\mathrm{bs}_f(x) = \mathrm{bs}(f)$. Assume without loss of generality that $f(x) = 0$. Let S_1 be the set of all 1-inputs with Hamming distance at least $(2/3)N$ from x. For any partial assignment p consistent with x, let $S_1^p \subseteq S_1$ be the set of all inputs y in S_1 that are consistent with p.

There are two cases. If S_1^p is non-empty for all partial assignments p consistent with x of size less than $\mathrm{bs}_f(x)/6$, then we can pick the promise to be $P = \{x\} \cup S_1$. It then follows that certifying that $f|_P$ is 0 on input x takes at least $\mathrm{bs}_f(x)/6$ queries, whence $\mathrm{R}_0(f|_P) \geq \mathrm{bs}_f(x)/6$. On the other hand, a randomized algorithm can make 1 query to check if the input differs from x. Thus $\mathrm{R}(f|_P) = 1$.

The other case is that there is some partial assignment p of size less than $\mathrm{bs}_f(x)/6$ such that S_1^p is empty. We restrict our attention to inputs consistent with p. Since x has $\mathrm{bs}_f(x)$ disjoint sensitive blocks, it has at least $(5/6)\,\mathrm{bs}_f(x)$ disjoint sensitive blocks that do not overlap with p. We exclude blocks of size larger than $N/3$. Since there are at most 2 such blocks, this leaves at least $(5/6)\,\mathrm{bs}_f(x) - 2$. Let B be the set of inputs we get by flipping one of these blocks of x. Then B contains only 1-inputs to f that are consistent with p, all of which have Hamming distance at most $N/3$ from x. Since $\mathrm{bs}_f(x) = \mathrm{bs}(f) > 6$, we have $B \neq \emptyset$.

Let S be the set of inputs consistent with p that have Hamming distance at least $(2/3)N$ from x. Since S_1^p is empty, S contains only 0-inputs to f. Let $P = B \cup S$. Now, consider certifying that an input y to $f|_P$ is a 1-input. Since all inputs of Hamming distance at least $(2/3)N$ from x that are consistent with p are 0-inputs, this requires showing at least $N/3 - |p|$ bits of y. Since $|p| < \mathrm{bs}_f(x)/6 \leq N/6$, this is at least $N/6$. Thus $\mathrm{R}_0(f|_P) \geq N/6 \geq \mathrm{bs}(f)/6$.

On the other hand, a bounded-error randomized algorithm can simply query a bit of the input at random, and check for agreement with x. If the bit agrees, the algorithm can output 1, and if the bit disagrees, the algorithm can output 0. This works because 0-inputs have distance at least $(2/3)N$ from x, while all 1-inputs have distance at most $N/3$ from x (since the sensitive blocks used to construct B were of size at most $N/3$). Thus $\mathrm{R}(f|_P) = 1$. ◀

7 Other Query Complexity Results

We can use some of the tools introduced in the previous sections to prove some new relations in query complexity. In Section 7.1, we prove a quadratic relationship between $\mathrm{D}(f)$ and $\mathrm{Q}(f)$ for partial functions f that have small domain. In Section 7.2, we prove a quadratic relationship between $\mathrm{D}(f)$ and $\mathrm{Q}(f)$ for total functions f for which $\mathrm{H}(\mathrm{C}_f)$ is small.

7.1 Query Complexity on Small Promises

We prove Theorem 1.3, which we restate for convenience.

▶ **Theorem 1.3.** *Let $f : \{0,1\}^N \to \{0,1\}$ be a partial function, and let $\mathrm{Dom}(f)$ denote the domain of f. Then*

$$\mathrm{Q}(f) = \Omega\left(\frac{\sqrt{\mathrm{D}(f)}}{\log|\mathrm{Dom}(f)|}\right).$$

Proof. We follow the proof of Theorem 3.2. The randomized algorithm used in that proof relies only on the existence of a randomized algorithm distinguishing a string $a \in \{0,1\}^N$ from either $f^{-1}(0)$ or $f^{-1}(1)$, which is in turn guaranteed by Lemma 3.1. To make that algorithm deterministic, we only need to turn this distinguishing algorithm into a deterministic one. By Lemma 5.1, we have $\mathrm{C}_f(x) = O(\mathrm{RC}_f(x) \log |\operatorname{Dom}(f)|)$. On the task of distinguishing a single input from a set of inputs, certificate complexity equals deterministic query complexity. Using this observation, we can modify the proof of Theorem 3.2 to get the result

$$\mathrm{D}(f) = O(\mathrm{Q}(f)^2 \operatorname{Bal}(f) \log |\operatorname{Dom}(f)|) = O(\mathrm{Q}(f)^2 \log^2 |\operatorname{Dom}(f)|),$$

from which the desired result follows. ◀

7.2 Relationship for Total Functions

We can use H-indices to improve some of the relationships between complexity measures on total functions, proving Theorem 1.5. Recall that for total functions, we have $\mathrm{D}(f) \leq \mathrm{C}(f) \operatorname{bs}(f)$ and $\operatorname{bs}(f) = O(\mathrm{Q}(f)^2)$, from which we have $\mathrm{D}(f) = O(\mathrm{Q}(f)^2 \mathrm{C}(f))$. We strengthen this result to $\mathrm{D}(f) = O(\mathrm{Q}(f)^2 \mathrm{H}(\sqrt{\mathrm{C}_f})^2)$ for total Boolean functions. Since $\mathrm{H}(\sqrt{\mathrm{C}_f}) \leq \sqrt{\mathrm{C}(f)}$, this result is always stronger. In addition, since $\mathrm{C}(\mathrm{OR}) = n$ and $\mathrm{H}(\mathrm{C}_{\mathrm{OR}}) = 1$, this improvement is sometimes very strong.

We restate Theorem 1.5 for convenience.

▶ **Theorem 1.5.** *Let $f : \{0,1\}^N \to \{0,1\}$ be a total function. Then*

$$\mathrm{D}(f) = O(\mathrm{Q}(f)^2 \mathrm{H}(\sqrt{\mathrm{C}_f})^2).$$

Proof. The proof follows the proof that $\mathrm{D}(f) \leq \mathrm{C}(f) \operatorname{bs}(f)$ [4]. We start by reviewing this proof. The deterministic algorithm repeatedly picks possible 0-certificates that are consistent with the input observed so far, and queries the entries of these certificates. If the queried entries match the 0-certificate, the algorithm is done (the value of $f(x)$ is known to be 0). If ever there are no additional 0-certificates consistent with the observed part of the input, the value of the function is known to be 1.

The key insight is that if this process repeats k times, then the block sensitivity of the function is at least k. Indeed, let p be the partial assignment revealed after k iterations. Pick a 1-input y for f that is consistent with p. Let B_i be the set of entries queried in the i-th iteration of the algorithm. Then for each i, there is a way to change only the variables in B_i to form a 0-certificate for f. It follows that each B_i contains a sensitive block for y. Since the B_i sets are disjoint, we get $\operatorname{bs}_f(y) \geq k$, so $\operatorname{bs}(f) \geq k$.

We modify the algorithm as follows. In each step, we only allow the algorithm to pick 0-certificates that are of size at most $\mathrm{H}(\sqrt{\mathrm{C}_f})^2$. Thus the algorithm uses at most $\operatorname{bs}(f) \mathrm{H}(\sqrt{\mathrm{C}_f})^2$ queries before it gets stuck. When it gets stuck, either the value of f on the input is determined, or else there are no more 0-certificates that are small enough.

Next, we repeat the same process with 1-certificates instead of 0-certificates. If the value of f is not yet determined, it means that the input is not consistent with any small enough certificates, so the certificate complexity of the input x is greater than $\mathrm{H}(\sqrt{\mathrm{C}_f})^2$. This gives $\sqrt{\mathrm{C}_f(x)} > \mathrm{H}(\sqrt{\mathrm{C}_f})$.

By definition of the H-index, there are now at most $2^{\mathrm{H}(\sqrt{\mathrm{C}_f})}$ possibilities for the input. We've therefore restricted f to a small domain P. We now use Theorem 1.3 to evaluate f using

$$O(\mathrm{Q}(f)^2 \log^2 |\operatorname{Dom}(f|_P)|) = O(\mathrm{Q}(f)^2 \mathrm{H}(\sqrt{\mathrm{C}_f})^2)$$

deterministic queries. This is added to the $\mathrm{bs}(f)\,\mathrm{H}(\sqrt{\mathrm{C}_f})^2$ queries from before. Using $\mathrm{bs}(f) = O(\mathrm{Q}(f)^2)$, we get

$$\mathrm{D}(f) = O(\mathrm{Q}(f)^2\,\mathrm{H}(\sqrt{\mathrm{C}_f})^2).$$ ◀

8 Sculpting in the Computational Complexity Model

In this section, we examine sculpting in the computational complexity model. We start with some notation. Given a language $L \subseteq \{0,1\}^*$, we let $L(x) \in \{0,1\}$ be its characteristic function. Also, given a language L together with a promise $P \subseteq \{0,1\}^*$, we let $L|_P$ be the promise problem of distinguishing the set $P \cap L$ from the set $P \setminus L$.

Now, we call the language L *sculptable* if there exists a promise P, such that the promise problem $L|_P$ is in PromiseBQP but not in PromiseBPP. We will use the following definition.

▶ **Definition 8.1** ([5])**.** A language L is called *paddable*, if there exists a polynomial-time function $f(x,y)$ such that

1. f is polynomial-time invertible, and
2. for all x, y, we have $x \in L \iff f(x,y) \in L$.

In other words, L is paddable if it is possible to "pad out"any input x with irrelevant information y, in an invertible way, without affecting membership in L.

The paddable languages were introduced by Berman and Hartmanis [5], as part of their exploration of whether all NP-complete languages are polynomial-time isomorphic: they showed that the answer was 'yes' for all *paddable* NP-complete languages. Under strong cryptographic assumptions, we now know that there exist NP-complete languages that are neither paddable nor isomorphic to each other [14]. Nevertheless, it remains the case that almost all the languages that "naturally arise in complexity theory" are paddable.

Next, let us say that PromiseBQP *is hard on average for* P/poly if there exists a promise problem $H|_S \in$ PromiseBQP, as well as a family of distributions $\{\mathcal{D}_n\}_n$ with support on the promise set S, such that

1. $\mathcal{D}_n$ is samplable in classical $\mathrm{poly}(n)$ time, and
2. there is no family of classical circuits $\{C_n\}_n$, of size $\mathrm{poly}(n)$, such that for all n,

$$\Pr_{y \sim \mathcal{D}_n}[C_n(y) = H(y)] \geq \frac{3}{4}.$$

So for example, because of Shor's algorithm [18], combined with the worst-case/average-case equivalence of the discrete log problem, we can say that *if discrete log is not in* P/poly, *then* PromiseBQP *is hard on average for* P/poly.

We now prove Theorem 1.6, which we restate here for convenience.

▶ **Theorem 1.6.** *Assume* PromiseBQP *is hard on average for* P/poly*. Then every paddable language outside of* BPP *is sculptable.*

Proof. Let L be a paddable language, and let f be the padding function for L. Also, let $H|_S$ be any problem in PromiseBQP that is hard on average for P/poly, and let $\{\mathcal{D}_n\}_n$ be the associated family of hard distributions. Then we need to construct a promise, $P \subseteq \{0,1\}^*$, such that the promise problem $L|_P$ is in PromiseBQP but not in PromiseBPP.

Our promise P will simply consist of all inputs of the form $f(x,y,a)$ such that $y \in S$ and

$$L(x) \equiv H(y) + a \pmod 2.$$

Here $a \in \{0,1\}$ is a single bit, which we think of as concatenated onto the end of y.

Clearly, $L|_P$ is in PromiseBQP: just invert f to extract the "comment"(y,a), then compute $H(y)+a \pmod 2$.

We need to show that $L|_P$ is not in PromiseBPP. Suppose by contradiction that it was, and let $\mathcal{A}$ be the algorithm such that $\mathcal{A}(x)=L(x)$ for all $x \in P$. Then we'll show how to either

1. decide L in BPP (with no promise), or
2. decide H in P/poly, with high probability over $\mathcal{D}_n$.

Given an arbitrary input $x \in \{0,1\}^n$, imagine we do the following: first sample $y \sim \mathcal{D}_n$, then run $\mathcal{A}$ on the inputs $f(x,y,0)$ and $f(x,y,1)$. There are two cases: first suppose

$$\mathcal{A}\left(f(x,y,0)\right) = \mathcal{A}\left(f(x,y,1)\right).$$

Now, *one* of the two inputs $f(x,y,0)$ and $f(x,y,1)$ must belong to P. If $f(x,y,0) \in P$, then $\mathcal{A}\left(f(x,y,0)\right) = L(x)$, while if $f(x,y,1) \in P$, then $\mathcal{A}\left(f(x,y,1)\right) = L(x)$. Either way, then, we have learned whether $x \in L$, and we know we have learned this.

Second, suppose

$$\mathcal{A}\left(f(x,y,0)\right) \neq \mathcal{A}\left(f(x,y,1)\right).$$

Then assuming $y \in S$:

$$\begin{aligned}
&x \in L, y \in H \Longrightarrow \mathcal{A}\left(f(x,y,0)\right)=1 \Longrightarrow \mathcal{A}\left(f(x,y,1)\right)=0,\\
&x \in L, y \notin H \Longrightarrow \mathcal{A}\left(f(x,y,1)\right)=1 \Longrightarrow \mathcal{A}\left(f(x,y,0)\right)=0,\\
&x \notin L, y \in H \Longrightarrow \mathcal{A}\left(f(x,y,1)\right)=0 \Longrightarrow \mathcal{A}\left(f(x,y,0)\right)=1,\\
&x \notin L, y \notin H \Longrightarrow \mathcal{A}\left(f(x,y,0)\right)=0 \Longrightarrow \mathcal{A}\left(f(x,y,1)\right)=1.
\end{aligned}$$

Thus, regardless of whether $x \in L$, we have learned whether $y \in H$, and again we know we have learned this.

Now suppose there were an input $x \in \{0,1\}^n$, such that running $\mathcal{A}$ as above told us whether $y \in H$ with probability more than (say) $1/2$ over the choice of $y \sim \mathcal{D}_n$. Then let C_n be a polynomial-size circuit that hardwires x, and that given an input $y \in S$:

- Simulates both $\mathcal{A}\left(f(x,y,0)\right)$ and $\mathcal{A}\left(f(x,y,1)\right)$.
- Outputs $H(y)$ whenever it successfully learns the value of $H(y)$.
- Guesses a hardwired value for $H(y)$ (whichever of $\{0,1\}$ is more probable) whenever it does not.

Then

$$\Pr_{y \sim \mathcal{D}_n}\left[C_n(y)=H(y)\right] \geq \frac{3}{4},$$

violating the assumption that no such circuit exists.

So we conclude that for every $x \in \{0,1\}^n$, we must instead learn whether $x \in L$ with probability at least $1/2$ over the choice of $y \sim \mathcal{D}_n$. This, in turn, means that by simply generating y's randomly until we succeed, we can decide L in PromiseBPP. ◀

Next, given a language $H \subseteq \{0,1\}^*$, we say H is BPP-*bi-immune* if neither H nor its complement $\overline{H}$ has any infinite subset in BPP. The notion of immunity was introduced by [9]. Here is a useful alternative characterization:

▶ **Lemma 8.2.** *A language H is* BPP*-bi-immune if and only if there is no infinite set $S \in$ BPP, such that the promise problem $H|_S$ is solvable in* PromiseBPP*.*

Proof. First, suppose H is not BPP-bi-immune, so that either H or $\overline{H}$ has an infinite subset $S \in \mathsf{BPP}$. Then clearly, S itself is an infinite set in BPP such that the promise problem $H|_S$ is trivial (the answer is either always 0 or always 1).

Conversely, suppose there exists an infinite set $S \in \mathsf{BPP}$ such that $H|_S$ is solvable in polynomial time. Then clearly $S \cap H$ and $S \cap \overline{H}$ are both in BPP, and at least one of the two must be infinite. So H is not BPP-bi-immune. ◀

We now suggest what, as far as we know, is a new conjecture in quantum complexity theory.

▶ **Conjecture 8.3.** *There exists a* BPP*-bi-immune language in* BQP*.*

Conjecture 8.3 is extremely strong. Note, in particular, that none of the "standard"BQP languages, such as languages based on factoring or discrete log, will be BPP-bi-immune, because they all have infinite special cases that are classically recognizable and easy (for example, the powers of 2, in the case of factoring). Nevertheless, we believe Conjecture 8.3 is plausible. As a concrete candidate for a BPP-bi-immune language in BQP, let $g : \{0,1\}^* \rightarrow \{0,1\}^*$ be some strong pseudorandom generator. Then consider the language

$$L = \{x : g(x), \text{ interpreted as an integer, has an odd number of distinct prime factors}\}.$$

We now prove Theorem 1.7, restated here for convenience.

▶ **Theorem 1.7.** *Assume Conjecture 8.3. Then every language outside of* BPP *is sculptable.*

Proof. Assume by way of contradiction that $L \notin \mathsf{BPP}$ is non-sculptable. Also, let H be a BPP-bi-immune language in BQP. Then consider the set

$$S := \{x : L(x) = H(x)\}.$$

By our assumption, S is a promise on which no superpolynomial quantum speedup is possible for L, and $\overline{S}$ is another such promise. Hence, there must be a BPP algorithm, call it $\mathcal{A}_S$, that solves the promise problem $H|_S$, which (by the definition of S) is equivalent to solving $L|_S$. And there must be another polynomial-time classical algorithm, call it $\mathcal{A}_{\overline{S}}$, that solves $H|_{\overline{S}}$, which (again by the definition of S) is equivalent to solving $\overline{L}|_{\overline{S}}$.

Now, given an input x, suppose we run both $\mathcal{A}_S$ and $\mathcal{A}_{\overline{S}}$. Then as in the proof of Theorem 1.6, there are two possibilities. If $\mathcal{A}_S(x) = \mathcal{A}_{\overline{S}}(x)$, then $x \in S$ implies $H(x) = \mathcal{A}_S(x)$ while $x \notin S$ implies $H(x) = \mathcal{A}_{\overline{S}}(x)$, so either way we have learned $H(x)$ (and we know that we have learned it). On the other hand, if $\mathcal{A}_S(x) \neq \mathcal{A}_{\overline{S}}(x)$, then $x \in S$ implies $L(x) = \mathcal{A}_S(x)$ while $x \notin S$ implies $L(x) = 1 - \mathcal{A}_{\overline{S}}(x)$. So, merely by seeing that $\mathcal{A}_S(x)$ and $\mathcal{A}_{\overline{S}}(x)$ are different, we have learned $L(x)$ (and we know that we have learned it).

In summary, there is a BPP algorithm $\mathcal{B}$ that, for every input $x \in \{0,1\}^*$, correctly outputs either $H(x)$ or $L(x)$, and that moreover tells us which one it output.

Now let Q be the set of all x such that $\mathcal{B}(x)$ outputs $H(x)$. Then there are two possibilities: if Q is finite, then $\mathcal{B}$ decides L on all but finitely many inputs. Hence $L \in \mathsf{BPP}$, contrary to assumption. If, on the other hand, Q is infinite, then $H|_Q$ is an infinite promise problem in PromiseBPP. So H was not BPP-bi-immune, again contrary to assumption. ◀

In Theorem 1.6 and Theorem 1.7, there is almost nothing specific to the complexity classes BQP and BPP, apart from some simple closure properties. Thus, one can prove analogous sculpting theorems for many other pairs of complexity classes. In some cases, we do not even need an unproved conjecture. For example, we have:

▶ **Theorem 8.4.** *For every language $L \notin \mathsf{P}$, there exists a promise S such that $L|_S$ is solvable in exponential time, but is not solvable in polynomial time.*

Proof. The proof of Theorem 1.7 follows through for P and EXP instead of BPP and BQP. In addition, it is known that there is a P-bi-immune language in EXP [5]. The desired result follows. ◀

9 Concluding Remarks and Open Problems

In this work, we gave a full characterization of the class of Boolean functions f that can be sculpted into a promise problem with an exponential quantum speedup in query complexity. We similarly characterized sculptability for R_0 vs. R and D vs. R_0. Along the way, we showed that Q is polynomially related (indeed, *quadratically* related) to D and R for a much wider set of promise problems than was previously known. Finally, we studied sculpting in *computational* complexity, giving a strong conjecture under which every language outside BPP is sculptable into a superpolynomial quantum speedup, and a weaker conjecture under which every *paddable* language outside BPP is sculptable.

One might object that many of our sculpted promise problems are somewhat artificial. This is particularly clear in the case of paddable languages, where (in essence) one uses the paddability to append to each instance x, as a "comment," an instance of a hard BQP problem (such as factoring) that is promised to have the same answer as x. Even in the query complexity setting, however, one can observe by direct analogy that *the property of being sculptable is not closed under the removal of dummy variables.* So for example, we saw before that the N-bit OR function is not sculptable. By contrast, observe that the function

$$f(x_1, \ldots, x_{2N}) := \mathrm{OR}(x_1, \ldots, x_N)$$

is sculptable. This follows as an immediate consequence of Theorem 1.1: just by adding dummy variables to the OR function, we have vastly increased the number of inputs x that have large certificate complexity, from 1 to 2^N. However, an even simpler way to see why f is sculptable, is that we can embed (say) Simon's problem into the variables $x_{N+1}, \ldots, x_{2N}$, and then impose the promise that

$$\mathrm{OR}(x_1, \ldots, x_N) = \textsc{Simon}(x_{N+1}, \ldots, x_{2N})$$

(in addition to the Simon promise itself).

Of course, most Boolean functions do *not* contain such dummy variables, so the problems of sculpting them, and deciding whether they are sculptable at all, are much more complicated, as we saw in this paper.

Now, it might feel like "cheating" to sculpt a promise problem with a large quantum/classical gap by using dummy variables to encode a different, unrelated problem. If so, however, that points to an interesting direction for future research: namely, can we somehow formalize what we mean by a "natural" special case of a problem, and can we then understand which problems are "naturally" sculptable?

Here are some more specific open problems.

- Some of our inequalities could be off by polynomial factors; it would be nice to tighten them (or prove separations). For example, it may be possible to improve Theorem 1.3 to $\mathrm{Q}(f) = \Omega(\sqrt{\mathrm{D}(f)/\log|\mathrm{Dom}(f)|})$, quadratically improving the $\log|\mathrm{Dom}(f)|$ factor.

- Can our results – and specifically, Theorem 1.5 – be used to improve the relation $\mathrm{D}(f) = O(\mathrm{Q}(f)^6)$ due to Beals et al. [4]?
- Can we give a characterization of the sculptable Boolean functions in *communication* complexity – analogous to this paper's characterization of sculptability in query complexity?
- Is there any natural pair of complexity classes $\mathcal{C} \subseteq \mathcal{D}$, for which $\mathcal{C}$ is known or believed to be strictly contained in $\mathcal{D}$, and yet it is plausible that no languages in $\mathcal{D}$ are $\mathcal{C}$-bi-immune, and (related to that) there exist languages $L \notin \mathcal{C}$ that *cannot* be sculpted into a promise problem in $\mathcal{D} \setminus \mathcal{C}$?
- One can, of course, consider sculpting for many other pairs of computational models, besides R vs. Q or R_0 vs. R or D vs. R_0. One interesting case is sculpting versus certificate complexity – for example, D vs. C. What is the correct characterization there?

We make some observations on the last problem. It's easy to see that $\mathrm{D}(\mathrm{OR}|_P) = \mathrm{C}(\mathrm{OR}|_P)$ for any promise P, so sculpting D vs. C is not always possible. On the other hand, sculpting D vs. C is sometimes possible even when $\mathrm{H}(\mathrm{C}_f)$ is small. To see this, consider the function f with $f(x) = 1$ if and only if the Hamming weight of x is 1, and the single '1' bit occurs on the left half of the input string. This function can be sculpted to $\mathrm{D}(f|_P) = N/2$ and $\mathrm{C}(f|_P) = 1$ by setting P to the set of inputs with Hamming weight 1. However, $\mathrm{H}(\mathrm{C}_f) = O(\log N)$ for this function, since all inputs with Hamming weight at least 2 have small certificates (just display two '1' bits).

This means something qualitatively different happens with D vs. C than what was found in this paper.

Acknowledgements. We thank Robin Kothari for many helpful discussions.

References

1 Scott Aaronson. Quantum certificate complexity. *SIAM Journal on Computing*, 35(4):804–824, 2006.

2 Scott Aaronson, Shalev Ben-David, and Robin Kothari. Separations in query complexity using cheat sheets. *arXiv preprint* `arXiv:1511.01937`, 2015.

3 Andris Ambainis, Kaspars Balodis, Aleksandrs Belovs, Troy Lee, Miklos Santha, and Juris Smotrovs. Separations in query complexity based on pointer functions. *arXiv preprint* `arXiv:1506.04719`, 2015.

4 Robert Beals, Harry Buhrman, Richard Cleve, Michele Mosca, and Ronald de Wolf. Quantum lower bounds by polynomials. *Journal of the ACM*, 48(4):778–797, 2001. `arXiv:arXiv:quant-ph/9802049`, `doi:10.1145/502090.502097`.

5 Leonard Berman and Juris Hartmanis. On isomorphisms and density of np and other complete sets. *SIAM Journal on Computing*, 6(2):305–322, 1977.

6 Harry Buhrman and Ronald de Wolf. Complexity measures and decision tree complexity: a survey. *Theoretical Computer Science*, 288(1):21–43, 2002. `doi:10.1016/S0304-3975(01)00144-X`.

7 Harry Buhrman, Lance Fortnow, Ilan Newman, and Hein Röhrig. Quantum property testing. *SIAM Journal on Computing*, 37(5):1387–1400, 2008.

8 Richard Cleve. The query complexity of order-finding. *Information and Computation*, 192(2):162–171, 2004.

9 Philippe Flajolet and Jean-Marc Steyaert. On sets having only hard subsets. In *Automata, Languages and Programming*, pages 446–457. Springer, 1974.

10 Jorge E Hirsch. An index to quantify an individual's scientific research output. *Proceedings of the National academy of Sciences of the United States of America*, 102(46):16569–16572, 2005.

11 Bo'az Klartag and Oded Regev. Quantum one-way communication can be exponentially stronger than classical communication. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 31–40. ACM, 2011.

12 Ilan Kremer. *Quantum communication*. PhD thesis, Citeseer, 1995.

13 Raghav Kulkarni and Avishay Tal. On fractional block sensitivity. *Electronic Colloquium on Computational Complexity (ECCC) TR13-168*, 2013.

14 Stuart A Kurtz, Stephen R Mahaney, and James S Royer. The isomorphism conjecture fails relative to a random oracle. *Journal of the ACM (JACM)*, 42(2):401–420, 1995.

15 Ran Raz. Exponential separation of quantum and classical communication complexity. In *Proceedings of the thirty-first annual ACM symposium on Theory of computing*, pages 358–367. ACM, 1999.

16 Norbert Sauer. On the density of families of sets. *Journal of Combinatorial Theory, Series A*, 13(1):145–147, 1972.

17 Saharon Shelah. A combinatorial problem; stability and order for models and theories in infinitary languages. *Pacific Journal of Mathematics*, 41(1):247–261, 1972.

18 Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, 1997. `arXiv:quant-ph/9508027`.

19 Andrew Chi-Chih Yao. Probabilistic computations: toward a unified measure of complexity. In *Annual Symposium on Foundations of Computer Science*, volume 17, page 222. Institute of Electrical and Electronics Engineers, 1977.

20 Bohua Zhan, Shelby Kimmel, and Avinatan Hassidim. Super-polynomial quantum speed-ups for boolean evaluation trees with hidden structure. In *Proceedings of the 3rd Innovations in Theoretical Computer Science conference*, pages 249–265. ACM, 2012.

A Properties of H Indices

▶ **Lemma A.1.** *Let* $g : \{0,1\}^n \to [0,\infty)$. *Define*

$$\mathrm{H}(g) := \inf\left\{h \in [0,\infty) : |\{x \in \{0,1\}^n : g(x) > h\}| \leq 2^h\right\}.$$

Then

1. $\mathrm{H}(g) \in [0,n]$
2. $\mathrm{H}(g) \leq \max_x g(x)$
3. *The number of* $x \in \{0,1\}^n$ *for which* $g(x) > \mathrm{H}(g)$ *is at most* $2^{\mathrm{H}(g)}$ *(equivalently, the infimum in the definition of* $\mathrm{H}(g)$ *is actually a minimum)*
4. *If* $g' : \{0,1\}^n \to [0,\infty)$ *is such that* $g(x) \leq g'(x)$ *for all* $x \in \{0,1\}^n$, *then* $\mathrm{H}(g) \leq \mathrm{H}(g')$
5. *If* $\alpha : [0,\infty) \to [0,\infty)$ *is an increasing function, then* $\mathrm{H}(\alpha \circ g) \leq \max\{\mathrm{H}(g), \alpha(\mathrm{H}(g))\}$.
6. *There are at least* $2^{\mathrm{H}(g)}$ *inputs* $x \in \{0,1\}^n$ *with* $g(x) \geq \mathrm{H}(g)$.

Proof. Let $S_g(h) = \{x \in \{0,1\}^n : g(x) > h\}$ and let $H_g = \{h \in [0,\infty) : |S_g(h)| \leq 2^h\}$. Then $\mathrm{H}(g) = \inf H_g$. Part 1 follows from noticing that for all h, $S_g(h) \subseteq \{0,1\}^n$, so $|S_g(h)| \leq 2^n$, whence $n \in H_g$. Part 2 follows from noticing that $S_g(\max_x g(x))$ is empty, so $\max_x g(x) \in H_g$.

To show 3, we show that H_g contains its infimum. Consider an infinite decreasing sequence $h_1, h_2, \ldots \in H_g$ that converges to $\mathrm{H}(g)$. Then the sequence $|S_g(h_1)|, |S_g(h_2)|, \ldots$ is a non-decreasing sequence of integers which is bounded above by 2^n. In addition, $S_g(h_i) \subseteq S_g(h_{i+1})$

for all i. It follows that there is some ℓ such that $S_g(h_i) = S_g(h_\ell)$ for all $i \geq \ell$. For each $x \in S_g(h_\ell)$, we have $g(x) > h_\ell > \mathrm{H}(g)$, and for each $x \notin S_g(h_\ell)$, we have $g(x) \leq h_i$ for all i. It follows that $g(x) \leq \mathrm{H}(g)$ for each $x \notin S_g(h_\ell)$, so $S_g(\mathrm{H}(g)) = \{x \in \{0,1\}^n : g(x) > \mathrm{H}(g)\} = S_g(h_i)$ for all $i \geq \ell$. Finally, since $h_i \in H_g$ for all i, we have $|S_g(\mathrm{H}(g))| = |S_g(h_i)| \leq 2^{h_i}$ for all $i \geq \ell$. From this it follows that $|S_g(\mathrm{H}(g))| \leq \lim_{i\to\infty} 2^{h_i} = 2^{\mathrm{H}(g)}$, so $\mathrm{H}(g) \in H_g$.

We now show 4. If g' is point-wise greater or equal to g, then $S_g(\mathrm{H}(g')) \subseteq S_{g'}(\mathrm{H}(g'))$. Since $\mathrm{H}(g') \subseteq H_g$, we have $|S_{g'}(\mathrm{H}(g'))| \leq 2^{\mathrm{H}(g')}$, so $|S_g(\mathrm{H}(g'))| \leq 2^{\mathrm{H}(g')}$. Thus $\mathrm{H}(g') \in H_g$, so $\mathrm{H}(g) = \inf H_g \leq \mathrm{H}(g')$.

We prove 5. Let α be an increasing function. We have

$$S_g(\mathrm{H}(g)) = \{x \in \{0,1\}^n : g(x) > \mathrm{H}(g)\} = \{x : \alpha \circ g(x) > \alpha(\mathrm{H}(g))\} = S_{\alpha\circ g}(\alpha(\mathrm{H}(g)).$$

Thus

$$|S_{\alpha\circ g}(\max\{\mathrm{H}(g), \alpha(\mathrm{H}(g))\})| \leq |S_{\alpha\circ g}(\alpha(\mathrm{H}(g))| = |S_g(\mathrm{H}(g))| \leq 2^{\mathrm{H}(g)} \leq 2^{\max\{\mathrm{H}(g),\alpha(\mathrm{H}(g))\}}$$

so $\max\{\mathrm{H}(g), \alpha(\mathrm{H}(g))\} \in H_{\alpha\circ g}$. Hence $\mathrm{H}(\alpha \circ g) \leq \max\{\mathrm{H}(g), \alpha(\mathrm{H}(g))\}$.

Finally, we show 6. If it was false, there would be less than $2^{\mathrm{H}(g)}$ inputs with $g(x) \geq \mathrm{H}(g)$. Thus there is some $\epsilon > 0$ such that there are less than $2^{\mathrm{H}(g)-\epsilon}$ inputs with $g(x) \geq \mathrm{H}(g) > \mathrm{H}(g) - \epsilon$. But this implies $\mathrm{H}(g) - \epsilon \geq \mathrm{H}(g)$, a contradiction. ◀

B Proof of Lemma 2.1

▶ **Lemma B.1.** *Let $S \subseteq \{0,1\}^N$ be a collection of strings. Then there is a shattered set of indices of size at least*

$$\frac{\log|S|}{\log(N+1)}.$$

Proof. Let d be the size of the largest set that is shattered by S. Then the Sauer-Shelah lemma [16] states

$$|S| \leq \sum_{i=0}^{d} \binom{N}{i}.$$

A well-known bound states

$$\sum_{i=0}^{d} \binom{N}{i} \leq 2^{\mathcal{H}(d/N)N},$$

where $\mathcal{H}(d/N)$ is the binary entropy of d/N. Then

$$\log_2|S| \leq \mathcal{H}(d/N)N = d\log_2(N/d) + (N-d)\log_2(1 + d/(N-d))$$

$$\leq d\log_2(N/d) + d\log_2 e = d\log_2 N - d\log_2(d/e) \leq d\log_2 N$$

(if $d \geq e$). Thus

$$d \geq \frac{\log|S|}{\log N}$$

unless $d \leq 2$.

The Sauer-Shelah lemma implies $|S| \leq 1$ when $d = 0$ and $|S| \leq N^2$ when $d = 2$ (assuming $N \geq 2$). The only problematic case is $d = 1$ and $|S| = N + 1$. Thus, in all cases, we have $d \geq \log|S|/\log(N+1)$, as desired. ◀

A Linear Time Algorithm for Quantum 2-SAT

Niel de Beaudrap[*1] **and Sevag Gharibian**[†2]

1 **Department of Computer Science, University of Oxford, Oxford, UK**
niel.debeaudrap@cs.ox.ac.uk

2 **Department of Computer Science, Virginia Commonwealth University, Richmond, USA**
sgharibian@vcu.edu

Abstract

The Boolean constraint satisfaction problem 3-SAT is arguably the canonical NP-complete problem. In contrast, 2-SAT can not only be decided in polynomial time, but in fact in deterministic linear time. In 2006, Bravyi proposed a physically motivated generalization of k-SAT to the quantum setting, defining the problem "quantum k-SAT". He showed that quantum 2-SAT is also solvable in polynomial time on a classical computer, in particular in deterministic time $O(n^4)$, assuming unit-cost arithmetic over a field extension of the rational numbers, where n is the number of variables. In this paper, we present an algorithm for quantum 2-SAT which runs in linear time, *i.e.* deterministic time $O(n+m)$ for n and m the number of variables and clauses, respectively. Our approach exploits the transfer matrix techniques of Laumann *et al.* [QIC, 2010] used in the study of phase transitions for random quantum 2-SAT, and bears similarities with both the linear time 2-SAT algorithms of Even, Itai, and Shamir (based on backtracking) [SICOMP, 1976] and Aspvall, Plass, and Tarjan (based on strongly connected components) [IPL, 1979].

1998 ACM Subject Classification F.2.1 Numerical Algorithms and Problems

Keywords and phrases quantum 2-SAT, transfer matrix, strongly connected components, limited backtracking, local Hamiltonian

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.27

1 Introduction

Boolean constraint satisfaction problems lie at the heart of theoretical computer science. Among the most fundamental of these is k-SAT, in which one is given a formula ϕ on n variables, consisting of a conjunction $\phi(x) = C_1 \wedge C_2 \wedge \cdots \wedge C_m$ of m clauses, each of which is a disjunction of k literals, *e.g.* $(x_h \vee \bar{x}_i \vee x_j)$ for $1 \leq h, i, j \leq n$. The problem is to determine whether there exists an assignment $x \in \{0,1\}^n$ which simultaneously satisfies all of the constraints C_i, *i.e.* for which $\phi(x) = 1$. While 3-SAT is NP-complete [6, 22, 16], 2-SAT admits a number of polynomial time algorithms (*e.g.* [7, 20, 10, 2, 23]), the fastest of which require just linear time [10, 2].

In 2006, Bravyi [3] introduced k-QSAT, a problem which generalizes k-SAT, as follows. In place of clauses C_i, acting on k-bit substrings of n bit strings $x \in \{0,1\}^n$, one considers

* Supported by Centrum Wiskunde & Informatica, European Commission FET-Proactive project Quantum Algorithms (QALGO) 600700, and the UK Quantum Technology Hub project NQIT.
† Supported by a Government of Canada NSERC Banting Postdoctoral Fellowship, the Simons Institute for the Theory of Computing at UC Berkeley, and NSF grant CCF-1526189. This project was initiated while SG was visiting Ronald de Wolf and Centrum Wiskunde & Informatica, whom SG thanks for their hospitality.

orthogonal projectors $\bar{\Pi}_i$ which act on k-qubit subsystems of an n-qubit system $|\psi\rangle \in \mathcal{H}^{\otimes n}$, where $\mathcal{H} := \mathbb{C}^2$. (A sketch of how k-SAT can be embedded into k-QSAT is given in Section 2.) These projectors extend to act on states $|\psi\rangle$ by defining $\Pi_i = \bar{\Pi}_i \otimes I$, so that Π_i acts as the identity on all tensor factors apart from those qubits on which $\bar{\Pi}_i$ is defined. One then considers $|\psi\rangle$ to "satisfy" the 2-QSAT instance if $\Pi_i |\psi\rangle = 0$ for all i. This formulation may be motivated, *e.g.*, by problems in many-body physics [9, 4]. While 3-QSAT is complete for QMA_1 [3, 13] (a quantum generalization[1] of NP), 2-QSAT is solvable in deterministic polynomial time [3], using $O(n^4)$ field operations over $\mathbb{C}$.

Given the existence of linear time algorithms for classical 2-SAT, this raises the natural question: *Can 2-QSAT also be solved in linear time?* Our main result in this paper is as follows.

▶ **Theorem 1.1.** *There exists a deterministic algorithm $SOLVE_{\mathrm{Q}}$ which, given an instance of 2-QSAT, outputs a representation of a satisfying assignment if one exists (presented as a list of one- and two-qubit unit vectors to be taken as a tensor product), and rejects otherwise.*

- *$SOLVE_{\mathrm{Q}}$ halts in time $O(n+m)$ on inputs on n qubits with m projectors (assuming unit-cost operations over $\mathbb{C}$).*
- *Furthermore, $SOLVE_{\mathrm{Q}}$ can produce its output using $O((n+m)M(n))$ bit operations, where $M(n)$ is the asymptotic upper bound on the cost of multiplying two n bit numbers;*
- *If the projectors are all product projectors, the algorithm $SOLVE_{\mathrm{Q}}$ requires only $O(n+m)$ bit operations regardless of what computable subfield $\mathbb{F} \subset \mathbb{C}$ the projector coefficients range over.*

In particular, the setting of product constraints above includes classical 2-SAT: in this case the bit-complexity of our algorithm matches optimal 2-SAT algorithms [10, 2].

▶ Remark. For general instances of 2-QSAT, the $O((m+n)M(n))$ bit-complexity of our algorithm compares favourably with the complexity of extracting a satisfying assignment using Bravyi's 2-QSAT algorithm, which requires $O(n^4 M(n))$ bit operations if one uses similar algebraic algorithms to ours. In "Significance and open questions" below, we discuss the question of field-operation-complexity vs. bit-complexity, as well as whether our algorithm is tight in terms of bit complexity.

Techniques employed

The origin of this work is the observation that Bravyi's 2-QSAT algorithm can be thought of as an analogue of Krom's 2-SAT algorithm [20], which involves computing the transitive closure of directed graphs. Krom's algorithm repeatedly applies a fixed inference rule for each pair of clauses sharing a variable. The repeated application of the inference rule leads to an $O(n^3)$ time to determine satisfiability and an $O(n^4)$ time to compute a satisfying assignment. Bravyi's algorithm has the same runtimes, measured in terms of the number of field operations.

This work aims to develop a quantum analogue of Aspvall, Plass, and Tarjan's (APT) linear time 2-SAT algorithm [2], which reduces 2-SAT to computing the strongly connected components of a directed graph. Note that classically $(\alpha \vee \beta)$ is equivalent to $(\bar{\alpha} \Rightarrow \beta)$ and $(\bar{\beta} \Rightarrow \alpha)$, for literals α and β. APT constructs an *implication graph* G of a 2-SAT instance

[1] Here, Quantum Merlin Arthur (QMA) is the quantum analogue of Merlin-Arthur (MA) in which the proof and verifier are quantum, and QMA_1 is QMA with perfect completeness. Unlike the classical setting, in which MA is known to admit perfect completeness [27, 12], whether QMA=QMA_1 remains open (see e.g. [15]).

ϕ, with vertices labelled by literals x_i and $\bar{x}_i$ for each i, and edges $\bar{\alpha} \to \beta$ and $\bar{\beta} \to \alpha$ for each clause $(\alpha \vee \beta)$. Then, they show that ϕ is satisfiable if and only if x_i and $\bar{x}_i$ are not in the same strongly connected component of G for any i [2]. As the strongly connected components of G can be computed in linear time [25], this yields a linear time algorithm for 2-SAT.

In the quantum setting, not all n-qubit states can be described by assignments to individual qubits (*e.g.*, entangled states). Fortunately, Chen *et al.* [4] show that we may reduce any instance of 2-QSAT to an instance which is satisfiable if and only if there is a satisfying state, in which qubits have separate assignments (see Section 2 for details). In this setting, there is a natural analogue of the equivalence $(x_i \vee x_j) \equiv (\bar{x}_i \Rightarrow x_j) \wedge (\bar{x}_j \Rightarrow x_i)$ in terms of so-called "transfer matrices" (e.g. [3, 21]). For any rank-1 quantum constraint $\Pi_{ij} \in \mathrm{L}\left(\mathbb{C}^2 \otimes \mathbb{C}^2\right)$ on qubits i and j, there exists a *transfer matrix* $\mathsf{T}_{ij} \in \mathrm{L}\left(\mathbb{C}^2\right)$, such that for any assignment $|\psi_i\rangle$ to qubit i such that $\mathsf{T}_{ij}|\psi_i\rangle \neq 0$, the state on qubit j for which the constraint Π_{ij} is satisfied is given by $\mathsf{T}_{ij}|\psi_i\rangle$.[2] (Conversely, for any $\mathsf{T}_{ij} \in \mathrm{L}\left(\mathbb{C}^2\right)$, there is a unique rank-1 orthogonal projector $\Pi_{ij} \in \mathrm{L}\left(\mathbb{C}^2 \otimes \mathbb{C}^2\right)$ whose nullspace is spanned by $|\psi_i\rangle \otimes \mathsf{T}_{ij}|\psi_i\rangle$ for $|\psi_i\rangle$ ranging over $\mathbb{C}^2$.) This suggests a quantum analogue G of an implication graph: For each possible assignment $|\psi\rangle$ to a qubit i, we define a vertex $(i, |\psi\rangle)$, and include a directed edge $(i, |\psi\rangle) \to (j, |\phi\rangle)$ if there is a transfer matrix T_{ij} (corresponding to some constraint Π_{ij}) such that $\mathsf{T}_{ij}|\psi\rangle = c|\phi\rangle$ for some $c \neq 0$. We then ask if for each qubit i, there is a vertex $(i, |\psi_i\rangle)$ which cannot reach any $(i, |\psi_i'\rangle)$ where $|\psi_i\rangle \not\propto |\psi_i'\rangle$. If there are such paths $(i, |\psi_i\rangle) \to \cdots \to (i, |\psi_i'\rangle)$ for all $|\psi_i\rangle$, this is analogous to x_i and $\bar{x}_i$ being in a common strong component in the APT algorithm.

As it stands, this approach has a shortcoming: In the quantum regime, each qubit has a *continuum* of possible assignments (rather than two), which may generate unbounded orbits in an APT-style algorithm. However, by applying techniques of Laumann *et al.* [21] from the study of phase transitions in random 2-QSAT, we may in some cases reduce the set of possible assignments for a qubit i to one or two. Consider the *interaction graph* G' of a 2-QSAT instance, in which vertices correspond to qubits, and two vertices are connected by an (undirected) edge if the corresponding qubits i and j are subject to a constraint Π_{ij}. Suppose $C = (v_1, \ldots, v_t, v_1)$ is a cycle in G', with transfer matrices $\mathsf{T}_{v_i v_{i+1}}$ arising from each constraint $\Pi_{v_i v_{i+1}}$, and compute $\mathsf{T}_C := \mathsf{T}_{v_t v_1} \cdots \mathsf{T}_{v_2 v_3} \mathsf{T}_{v_1 v_2}$. If T_C has a non-degenerate spectrum, then the only possible satisfying assignments for v_1 are eigenvectors of T_C [21] (see also Lemma 2.2). In effect, computing T_C "simulates" uncountably many (!) traversals $(i, |\alpha\rangle) \to \cdots \to (i, |\beta\rangle)$ in G; restricting to the eigenvectors of T_C corresponds to ignoring vertices in G which are infinitely far from the top of any topological order of G. If we hence describe cycles C with non-degenerate T_C as *discretizing*, this suggests the approach of finding a discretizing cycle at each qubit i, and using it to reduce the number of possible states on i to one or two. This simple principle is the starting point of our work.

Despite this simplicity, some obstacles must be addressed to obtain a linear-time algorithm. In the setting of random 2-QSAT [21], every cycle C is a discretising cycle with probability one, as there is zero probability that either a transfer matrix is singular, or that a product of them has a degenerate spectrum. This allows one to quickly reduce the space of assignments possible for a qubit. In contrast, in our setting (*i.e.*, worst case analysis), we cannot assume

[2] The usual convention is to describe quantum states by unit vectors in $\mathbb{C}^2$, albeit up to equivalence under multiplication by $z \in \mathbb{C}$ for $|z| = 1$. However, vectors produced via transfer matrices might not be normalised. As we are not explicitly concerned with the probabilities of any measurement outcomes obtained from quantum processes, we represent quantum states by vectors which are equivalent up to multiplication by arbitrary (non-zero) scalar factors.

such a distribution of transfer matrices arising from a 2-QSAT instance. For instance, any constraint Π_{ij} corresponding to a product operator (*e.g.*, a classical 2-SAT constraint) has a singular transfer matrix, which when multiplied with other singular matrices may give rise to a singular cycle matrix. Even if a discretising cycle C does exist using some of the edges jk, $k\ell$, ..., we may have to traverse those edges multiple times to discover C, which is worrisome for a linear-time algorithm. Furthermore, we must address the case in which there are no discretising cycles at all to get a discrete algorithm started. In order to demonstrate a *linear*-time algorithm for 2-QSAT in the spirit of APT, these problems must be carefully addressed.

Our approach to resolve these issues is as follows. In an instance of 2-QSAT in which all transfer matrices are non-singular, we show that discretising cycles are easy to find if they exist, and that the absence of discretising cycles allows one to easily obtain a satisfying state. If, on the other hand, singular transfer matrices are present, the corresponding product constraints $\Pi_{ij} = |\alpha\rangle\langle\alpha|_i \otimes |\beta\rangle\langle\beta|_j$ *themselves* impose a different discretising influence: If $|\alpha^\perp\rangle$ and $|\beta^\perp\rangle$ are states orthogonal to $|\alpha\rangle$ and $|\beta\rangle$ respectively, then at least one of the assignments $(i, |\alpha^\perp\rangle)$ or $(j, |\beta^\perp\rangle)$ is required for a satisfying assignment. This leads us to adopt an approach of "trial assignments" which is highly reminiscent of another linear-time 2-SAT algorithm due to Even-Itai-Shamir [10], which attempts to reduce to an instance of 2-QSAT with fewer product constraints by determining partial assignments satisfying Π_{ij}. (For simplicity, we also adopt the approach of trial assignments for qubits whose state-space have been reduced by discretizing cycles.) This leads us to our algorithm SOLVE_Q (Figure 1, in Section 4), which combines elements of both the Even-Itai-Shamir [10] and Apsvall-Plass-Tarjan [2] linear-time 2-SAT algorithms as described above.

Our approach can be summarised as follows. Following Chen *et al.* [4], we first preprocess our input instance Π and either determine that Π is unsatisfiable, or obtain a new 2-QSAT instance Π' which is satisfiable by a product state if Π is satisfiable at all. From this point on, our algorithm uses the central notion of a *chain reaction* (CR) (see Section 3): this roughly models the idea that given an assignment $|\psi_i\rangle$ to qubit i, following a sequence of transfer operators according to the implication graph of Π' deterministically results in assignments to a subset of other qubits in the instance. In particular, what we are interested in is finding *conflict-free* CRs, which are CRs that terminate without reassigning a value to a qubit j which conflicts with a previous assignment for j. To exploit conflict-free CRs, we first show a Set-and-Forget Theorem (Theorem 3.6), which essentially says the following: if Π' is satisfiable, then any choice of assignments to a subset S which is prescribed by a conflict-free CR, is also consistent with a global satisfying assignment. Thus, given such a conflict-free CR, we can remove the qubits in S and all constraints acting on it from Π', reducing to a smaller 2-QSAT instance Π'' which is satisfiable if and only if Π' is. Hence, the problem of deciding Π is reduced to the task of repeatedly finding conflict-free CRs. To show that the discovery of conflict-free CRs may be done in *linear* time, we use three key ideas. First, for any product constraint in the graph, there are two associated CRs C_1 and C_2; we show that at least one of these must be conflict-free, or Π is not satisfiable. Second, once all product constraints have been exhausted, our next source of conflict-free CRs is the notion of discretizing cycles. In general, it is *not* true that running a depth-first search in the constraint graph of Π'' will yield a discretizing cycle, even if such a cycle exists! However, we show that if all constraints are *entangled*, then a single depth-first search (per connected component of the interaction graph) indeed suffices to find the discretizing cycle. Third and finally, if no discretizing cycles exist in Π'', then we show that it is easy to find a conflict-free CR. The resulting algorithm, SOLVE_Q, is presented in Figure 1.

Previous work

There is a long history of polynomial time solutions for classical 2-SAT [24, 7, 20, 10, 2, 23], ranging from time $O(n^4)$ to $O(n+m)$. As we mention above, the most relevant of these to our setting are the algorithms of Even, Itai, and Shamir [10] (based on limited backtracking) and Aspvall, Plass, Tarjan [2] (based on strongly connected component detection).

In contrast, little work has been performed in the quantum setting. Until recently, Bravyi's algorithm was the only explicitly articulated algorithm for 2-QSAT, and requires $O(n^4)$ field operations and $O(n^4 M(n))$ bit operations. Other work on 2-QSAT instead concerns either the structure of the solution space of instances of 2-QSAT [21, 9, 4], or bounds on counting complexity [14, 8].

Propagation of assignments using transfer matrices is present already in Bravyi [3], and the results of Laumann *et al.* [21] allow us to restrict the possibly satisfying states on single qubits by finding discretising cycles. We incorporate these into efficient discrete algorithms for testing possible assignments, and provide a cost analysis in terms of field operations and bit operations. In contrast to the random 2-QSAT setting of [21], we do not assume any particular distribution on constraints.

Note: Very recently, Arad *et al.* [1] independently and concurrently presented an algorithm for 2-QSAT, which also runs in $O(n+m)$ time using unit-cost field operations. The overall structure of our algorithm appears similar to theirs, though our treatment of the key issue of 2-QSAT instances with only entangled constraints appears to use different techniques (in particular, Ref. [1] appears to be based on results of Ji, Wei, Zeng [14] which modify the instance itself, whereas we use ideas of [21] to tackle the existing instance via the concept of discretizing cycles). As well as obtaining an upper bound on field operations matching Ref. [1], we also include an analysis of the bit complexity of our algorithm $\mathrm{SOLVE}_{\mathrm{Q}}$, and in particular indicate how our algorithm matches the asymptotic bit complexity of the best algorithms on classical instances of 2-SAT.

Significance and open questions

From a complexity theoretic perspective, just as k-SAT and MAX-k-SAT are canonical NP-complete problems, Quantum k-SAT and its optimization variant, k-LOCAL HAMILTONIAN [19], are canonical QMA_1- and QMA-complete problems for $k \geq 3$ and $k \geq 2$ respectively [3, 13, 19, 17], thus making their study central to quantum complexity theory. From a many-body physics perspective, quantum k-SAT deals with the study of ground states of *frustration-free* local Hamiltonian systems. Such systems include Kitaev's well-known Toric code Hamiltonian [18] (which is 4-local), whose ground space encodes logical qubits of a topological quantum error correcting code. Our work can hence be viewed as aiming to understand which classical techniques for k-SAT can be generalized to explore the ground spaces of such frustration-free systems.

Bit complexity. We now discuss the number of field operations used by our algorithm, $O(m+n)$, versus the number of bit operations, $O((m+n)M(n))$, in Theorem 1.1. There is no such distinction in the complexity of existing 2-SAT algorithms: As bits have only a finite range of values, traversing a chain of implications in the implication graph poses no precision issues. In the quantum setting, however, such a traversal involves computing products of $O(n)$ transfer matrices over some field extension of the rationals. Trial assignments resulting from these products may require $O(n)$ bits per entry to represent; testing whether two possible assignments are equivalent may involve multiplying pairs of n-bit integers. This is the source

of the $M(n)$ term in the bit complexity estimate of Theorem 1.1. To compare, similar considerations applied to Bravyi's 2-QSAT algorithm gives an upper bound of $O(n^4 M(n))$ bit operations.

It is not obvious that a faster runtime in terms of bit complexity should be possible in general. As we show in Section 6, it is simple to construct a 2-QSAT instance with $m \in O(n)$ and whose unique product state solution requires $\Theta(n^2)$ bits to write down. Thus, among algorithms which explicitly output the entire solution, our algorithm is optimal up to log factors, taking time $O(nM(n)) \in \widetilde{O}(n^2)$ for $M(n) \in O(n \log(n)\, 2^{O(\log^*(n))})$ [11]. Furthermore, as we also show in Section 6, for any algorithm **A** for 2-QSAT which produces the marginal of a satisfying solution (if one exists) on a single qubit in reduced terms[3], there is a linear-time reduction from multiplication of n bit integers to the problem solved by **A**. It follows that such an algorithm **A** must run in time $\Omega(M(n))$. As discussed in Section 6, this implies that unless $M(n) \in O(n)$, there is no general algorithm for 2-QSAT with linear bit complexity if the output is required to be in reduced form.

Linear bit complexity. Theorem 1.1 gives a setting in which our algorithm does have linear bit complexity – when all constraints are product operators. This special case still has essentially quantum features, such as satisfiable instances requiring two-qubit entanglement (which our algorithm treats using techniques described in Section 2), and phase-transitions for satisfiability and counting complexity in randomly sampled instances which match those of 2-QSAT rather than classical 2-SAT [8]. It also includes the classical 2-SAT instances, for which our algorithm has optimal bit-complexity.

Open questions. Our algorithm uses results of Chen *et al.* [4], which shows that any satisfiable instance of 2-QSAT has a solution which is "almost" a product state (our algorithm finds such solutions). In the degenerate case, however, there may also exist satisfying states with long-range entanglement, which may also be of interest to find. As our aim here is to study the optimal computational complexity of 2-QSAT, as opposed to seeking particular types of solutions, we leave this as an open question. We also ask: Is the bit-complexity of $O((n+m)M(n))$ for producing explicit assignments optimal? Is there an $O(M(n))$ upper bound for producing representations of marginals of satisfying assignments?

Organization of this paper

In Section 2, we give notation, definitions, and the basic framework for our analysis (including transfer matrices). Section 3 presents a series of lemmas and theorems to demonstrate how to overcome the obstacles presented in this introduction, and which form the basis of a proof of correctness for our algorithm SOLVE_Q. Section 4 states SOLVE_Q. Section 5 sketches bounds on the runtime of SOLVE_Q in terms of the field operations and bit operations. Additional technical details are deferred to the full version of this paper. Section 6 discusses lower bounds on the bit complexity of 2-QSAT.

2 Preliminaries

We begin by setting notation, stating definitions, and laying down the basic framework for our algorithm, including details on transfer matrices.

[3] N.B. Our algorithm SOLVE_Q is not such an algorithm, as the output may include cancellable factors in its representation.

Notation

The notation := denotes a definition and $[n] := \{1, \ldots, n\}$. The vector space of (possibly non-normalised) single-qubit pure states is denoted $\mathcal{H} := \mathbb{C}^2$. For a string $x = x_1 x_2 \cdots x_n \in \{0,1\}^n$, we write $|x\rangle := |x_1\rangle \otimes \cdots \otimes |x_n\rangle$. For a vector space $\mathcal{X}$ over $\mathbb{C}$, we write $\mathrm{L}(\mathcal{X})$ for the set of linear operators on $\mathcal{X}$. The nullspace of an operator A is denoted $\ker(A)$. For vectors $|\psi\rangle$ and $|\phi\rangle$, we write $|\psi\rangle \propto |\phi\rangle$ if $|v\rangle = c\,|w\rangle$ for *non-zero* $c \in \mathbb{C}$; if we wish to also allow $c = 0$, we write $|\psi\rangle \propto^* |\phi\rangle$ instead. The latter two definitions extend straightforwardly to matrices. Given $|\psi\rangle \in \mathcal{H}$, we write $|\psi^\perp\rangle$ for the unique vector (up to scalar factors) which is orthogonal to $|\psi\rangle$.

2.1 Quantum 2-SAT

We now present a formal definition of quantum k-SAT (or k-QSAT).

▶ **Definition 2.1** (Quantum k-SAT [3])**.** Let $n \geq k$ be an integer, and $\{\Pi_i\}_{i=1}^m \subset \mathrm{L}\left(\mathcal{H}^{\otimes k}\right)$ be a set of k-local orthogonal projection operators (*i.e.*, of the form $I \otimes \bar{\Pi}_i$ for k-qubit projectors $\bar{\Pi}_i$) with coefficients over some number field $\mathbb{F}$.

Decision problem. Does there exist a state $|\psi\rangle \in \mathcal{H}^{\otimes n}$ such that $\Pi_i\,|\psi\rangle = 0$ for all $i \in [m]$?

Search problem. Produce a description of such a state $|\psi\rangle$ if it exists.

For precision reasons, we require in particular that the coefficients are drawn from a number field (a finite-degree field extension $\mathbb{F} = \mathbb{Q}[\omega]$). We suppose that $\mathbb{F}$ is also specified as part of the input by means of a minimal polynomial $p \in \mathbb{Q}[x]$ for which $\mathbb{F} \cong \mathbb{Q}[x]/p$, together with a specification of how $\mathbb{F}$ embeds into $\mathbb{C}$ [5]. (More details are given in the full version, where the runtime of the algorithm is carefully analyzed.) In the literature for 2-QSAT, one is usually more interested in how the structure of the placement of the projectors Π_i affects the solution space, rather than the complexity of the specification of $\mathbb{F}$ or the coefficients. We therefore suppose that there is some constant K which bounds from above the size of the specification of $\mathbb{F}$, and of the coefficients of the operators Π_i.

We next sketch how a 2-SAT instance ϕ can be embedded into 2-QSAT (cases $k > 2$ are similar). For each clause C on boolean variables (x_a, x_b), we define an operator $\Pi_C \in \mathrm{L}\left(\mathcal{H}^{\otimes 2}\right)$ of the form $\Pi_C := |c_a\rangle\langle c_a| \otimes |c_b\rangle\langle c_b|$, where $c_a = 1$ if the variable x_a is negated in C, and $c_a = 0$ otherwise; we fix c_b similarly. Then Π_C is satisfied by $|x_a x_b\rangle \in \mathcal{H}^{\otimes 2}$ if and only if C is satisfied by $x_a x_b \in \{0,1\}^2$. We extend Π_C to an operator on $\mathcal{H}^{\otimes n}$ by taking its tensor product with $I_2 \in \mathrm{L}\left(\mathcal{H}^{\otimes n-2}\right)$ on all tensor factors i apart from $a, b \in [n]$. Performing this for all clauses yields an instance of 2-QSAT, $\{\Pi_C\}$, in which all of the projectors are product operators (as mentioned in Section 1), and which imposes the same constraints on standard-basis vectors $|x\rangle$ as the clauses C impose on $x \in \{0,1\}^n$. Furthermore, as each Π_C is positive semidefinite and diagonal, any $|\psi\rangle$ for which $\Pi_C\,|\psi\rangle = 0$ for all clauses C must be a linear combination of vectors $|x\rangle$ which also satisfy $\Pi_C\,|x\rangle = 0$ for all C. Thus this instance of 2-QSAT is satisfiable if and only if the original instance of 2-SAT is, in which case there is a bijection between the solution space of the 2-SAT instance and a basis for the solution-space of the 2-QSAT instance.

Finally, for a given 2-QSAT instance, we denote by G its (potentially infinite) *implication graph* (defined in Section 1), and by G' its *interaction graph*, whose vertices are labelled by qubits, and with a distinct edge between vertices i, j for each projector acting on them.

Reduction to cases satisfied by product states

We mainly consider product-state solutions to instances of 2-QSAT, in spite of instances (such as those described in "Significance and open questions" in Section 1) in which no product state can be a solution. A paradigmatic example is given by a single constraint $\Pi_* = I_4 - |\Psi^-\rangle\langle\Psi^-|$, where $|\Psi^-\rangle := (|01\rangle - |10\rangle)/\sqrt{2}$; the unique satisfying assignment is the entangled state $|\Psi^-\rangle$. Chen *et al.* [4] nevertheless show that all instances of 2-QSAT are "almost" product-satisfiable in the following sense: The only pairs of qubits (i, j) which are entangled for all satisfying states are those for which the sum of all constraints on (i, j) is an operator S_{ij} of rank 3 (as with Π_* above). We may treat such pairs by imposing the unique assignment $|\psi_{ij}\rangle \in \ker(S_{ij})$, and considering what restrictions this imposes on other qubits k as a result of constraints on (i, k) or (j, k). If we find no conflicts as a result of all such assignments, we obtain a sub-problem which is either unsatisfiable, or satisfiable by a product state. (We describe this reduction in more detail in Section 4.)

Reduction to rank-1 instances

We may require that all constraints have rank 1 (but possibly with multiple constraints on pairs of qubits), by decomposing projectors Π_{ij} of higher rank into rank-1 projectors $\Pi_{ij,1}$, $\Pi_{ij,2}$, ..., for which $\Pi_{ij} = \sum_k \Pi_{ij,k}$. By the preceding reduction to product-satisfiable constraints, there will then be at most two independent constraints acting on any pair (i, j).

2.2 Transfer matrices

A central tool in this work is the *transfer matrix*, which for product states generalizes the equivalence between $(x_i \vee x_j)$ and $(\bar{x}_i \Rightarrow x_j) \wedge (\bar{x}_j \Rightarrow x_i)$ for bits. Consider a rank-1 constraint $\Pi_{ij} = |\phi\rangle\langle\phi|$ on qubits i and j, where $|\phi\rangle$ has Schmidt decomposition $|\phi\rangle = \alpha\,|a_0\rangle\,|b_0\rangle + \beta\,|a_1\rangle\,|b_1\rangle$. Then, the *transfer* matrices $\mathsf{T}_{\phi,ij}, \mathsf{T}_{\phi,ji} \in \mathrm{L}\left(\mathbb{C}^2\right)$ from i to j and from j to i are respectively given by:

$$\mathsf{T}_{\phi,ij} = \beta\,|b_0\rangle\langle a_1| - \alpha\,|b_1\rangle\langle a_0|\,, \qquad \mathsf{T}_{\phi,ji} = \beta\,|a_0\rangle\langle b_1| - \alpha\,|a_1\rangle\langle b_0|\,. \tag{1}$$

(When the state $|\phi\rangle$ is clear from context, we simply write T_{ij} and T_{ji}.) Given any assignment $|\psi_i\rangle \in \mathbb{C}^2$ on qubit i, the transfer matrix $\mathsf{T}_{\phi,ij}$ prescribes which single-qubit states $|\psi_j\rangle$ on j are required to satisfy Π_{ij}, via the constraint $|\psi_j\rangle \propto^* \mathsf{T}_{\phi,ij}\,|\psi_i\rangle$. If $\mathsf{T}_{\phi,ij}\,|\psi_i\rangle \neq 0$, then $|\psi_j\rangle$ is uniquely determined (up to equivalence by a scalar factor). This is guaranteed when $|\phi\rangle$ has Schmidt rank 2, as $\mathsf{T}_{\phi,ij}$ then has full rank. On the other hand, if $\mathsf{T}_{\phi,ij}\,|\psi_i\rangle = 0$, then Π_{ij} is satisfied for any assignment on j, so that j remains unconstrained. This situation may only occur if $|\phi\rangle$ is a product constraint, so that $\mathsf{T}_{\phi,ij}$ has a nullspace of dimension 1. This generalises the effect in the classical setting, that assigning $x_i := 1$ satisfies the constraint $C = (x_i \vee x_j)$, regardless of the value of x_j: the corresponding constraint and transfer matrix are $|\phi\rangle = |00\rangle$ and $\mathsf{T}_{\phi,ij} = -\,|1\rangle\langle 0|$, respectively.

Walk and cycle matrices

We take the closure of the transfer matrices, under composition along walks in the graph. For any walk $W = (v_1, v_2, \ldots v_k)$ in a graph $G = (V, E)$, multiplying the transfer matrices $\mathsf{T}_{v_{k-1}v_k} \cdots \mathsf{T}_{v_2v_3}\mathsf{T}_{v_1v_2}$ yields a new transfer matrix T_W, which we call the *walk matrix* of W (or *path matrix*, if W is a path). For such a walk W, define $W^{\mathsf{R}} := (v_k, v_{k-1}, \ldots, v_2, v_1)$. If a transfer matrix T_W has singular value decomposition $\mathsf{T}_W = s_0\,|\ell_0\rangle\langle r_0| + s_1\,|\ell_1\rangle\langle r_1|$, one may

show by induction on the length of W that

$$\mathsf{T}_{W^{\mathsf{R}}} = \pm\big(s_0 \,|r_1\rangle\langle\ell_1| + s_1 \,|r_0\rangle\langle\ell_0|\big), \tag{2}$$

where the sign depends on whether W has odd or even length. In particular, this implies that $\mathsf{T}_W \mathsf{T}_{W^{\mathsf{R}}} = \pm s_0 s_1 I$. Thus $\mathsf{T}_W \mathsf{T}_{W^{\mathsf{R}}} \propto^* I$ for all walks W, with a proportionality factor of zero if and only if T_W is singular. In particular, walk operators can sometimes be composed to represent "cancellation" of edges: For walks $U_1 = W'W$ and $U_2 = W^{\mathsf{R}}W''$, if T_W is invertible, we have $\mathsf{T}_{W'W''} \propto \mathsf{T}_{W''}\mathsf{T}_{W^{\mathsf{R}}}\mathsf{T}_W\mathsf{T}_{W'} = \mathsf{T}_{U_1U_2}$, representing a form of composition of walks in which repeated edges $(ij)(ji)$ cancel.

For $C = (v, u_1, u_2, \ldots, u_k, v)$ a cycle in G, the *cycle matrix of C at v* is just the walk operator T_C arising from the walk from v to itself along C. We consider the cycles C and (e.g.) $C' = (u_1, u_2, \ldots, u_k, v, u_1)$ to be distinct as walks; in particular, C and C' may give rise to distinct cycle matrices $\mathsf{T}_{C'} \not\propto \mathsf{T}_C$, which in any case represent operators on the state-spaces of distinct qubits.

Walk operators (and cycle operators in particular) allow us to more easily express long-range constraints implicit in the original projectors Π_{ij} (as one may show by induction):

▶ **Lemma 2.2** (Inconsistency Lemma). *Let $W = (v, v_1, v_2, \ldots, v_\ell, w)$ be a walk in G' with walk operator T_W, and let $|\Psi\rangle \in \mathcal{H}^{\otimes n}$ be a product of single-qubit states $|\psi_v\rangle$ for each $v \in [n]$. If $|\psi_w\rangle \not\propto^* \mathsf{T}_W |\psi_v\rangle$, then at least one constraint Π_{ij} corresponding to an edge in W is not satisfied by $|\Psi\rangle$.*

3 Efficient reductions via trial assignments in 2-QSAT

As outlined in Section 2, we consider rank-1 instances of 2-QSAT which either have a product solution or are unsatisfiable. In this section, we describe a means to incorporate transfer matrices into an efficient algorithm for 2-QSAT via the notion of a *chain reaction*: An EIS-style subroutine for trial assignments.

As in Section 1, we define the *implication graph* of a 2-QSAT instance to be an (infinite) directed graph $G = (V, E)$, where V is the set of pairs $(i, |\psi\rangle)$ for qubits i and (distinct) states $|\psi\rangle \in \mathcal{H}$. There is a directed edge $(i, |\psi\rangle) \to (j, |\phi\rangle)$ if and only if there is a constraint Π_{ij} with transfer matrix T_{ij} such that $\mathsf{T}_{ij} |\psi\rangle \propto |\phi\rangle$. A "chain reaction" is a depth-first exploration of the nodes of G:

▶ **Definition 3.1** (Chain reaction (CR)). For a qubit i and state $|\psi_i\rangle \in \mathcal{H}$, to *induce a chain reaction (CR) at i with $|\psi_i\rangle$* means to "partially traverse" G, starting from $(i, |\psi_i\rangle)$ and keeping a record of the vertices $(u, |\psi_u\rangle)$ seen for each u. This traversal is governed by a depth-first search (DFS) in the interaction graph G', as follows. For each vertex $(u, |\psi_u\rangle)$ visited and each edge $\{u, v\}$ in G', compute $\mathsf{T}_{uv} |\psi_u\rangle$. If this vector is non-zero, let $|\psi_v\rangle := \mathsf{T}_{uv} |\psi_u\rangle$, and traverse to $(v, |\psi_v\rangle)$ in G. For any vertex $(v, |\psi_v\rangle)$ visited by the CR, we say that the CR *assigns $|\psi_v\rangle$ to v*. In the sequence of vertices in G visited by the CR, we may refer to instances of vertices $(v, |\psi\rangle)$ for a given $v \in V$ as the *first assignment*, the *second assignment*, *etc.* made to v by the CR.

Edges of G' (and walks in G') which are traversed by the depth-first search (DFS) governing a chain reaction, are also said to be traversed by the chain reaction (CR) itself.

The role of CRs in our analysis is to reveal constraints imposed by transfer matrices in an efficient manner. Specifically, if the DFS in G' which governs the CR encounters a cycle, it will visit a vertex v in G' twice, and so makes "assignments" to v more than once. If these

assignments do not match, we say the CR has a *conflict*. If no such conflicts occur, the CR is called *conflict-free*. (In either case, it does not continue the traversal of the CR from the second, third, *etc.* assignments.) We formalise the intuitive significance of conflicts as follows:

▶ **Lemma 3.2** (Conflict Lemma). *If a CR induced at v with $|\psi_v\rangle \in \mathcal{H}$ has a conflict, then no product state $|\Psi\rangle \in \mathcal{H}^{\otimes n}$ for which the state of v is $|\psi_v\rangle$ is a satisfying assignment.*

Proof. A conflict in the CR indicates the presence of two walks W_1 and W_2 in the interaction graph G', from v to some vertex w, for which $\mathsf{T}_{W_1}|\psi_v\rangle \not\propto^* \mathsf{T}_{W_2}|\psi_v\rangle$. It follows from the Inconsistency Lemma (Lemma 2.2) that any product state in which v takes the state $|\psi_v\rangle$ is not satisfying. ◀

With the concept of CRs in hand, we can present the key ideas used by our algorithm. First, conflict-free CRs yield partial assignments, which preserve the satisfiability of the instance defined on the remaining unassigned qubits. Second, if a 2-QSAT instance is satisfiable, then a conflict-free CR can be found efficiently. Our algorithm (presented in Figure 1) essentially operates by repeatedly finding conflict-free CRs, and removing the qubits given assignments by each CR, until either a conflict is detected (in which case we reject), or no unassigned qubits remain (in which case we accept).

3.1 Using conflict-free chain reactions to remove qubits

The main result in this Section is Theorem 3.6 (Set-and-Forget Theorem), which is essentially the converse of Lemma 3.2, and allows us to reduce instances of 2-QSAT by providing partial solutions obtained from a CR induced on a single qubit.

We begin by proving a correspondence between CRs and walk operators, in the sense that if there is a walk $W = (v, v_1, v_2, \ldots, w)$ in G', and if $|\psi_v\rangle \notin \ker(\mathsf{T}_W)$, a CR induced at v with a state $|\psi_v\rangle$ should assign $\mathsf{T}_W|\psi_v\rangle$ to w. The obstacle here is that the CR might not traverse any of the edges of W before assigning a state to w; we must then relate W to other walks in G'. We do so as follows.

▶ **Lemma 3.3** (Unique Assignment Lemma). *Suppose there exists a state $|\psi\rangle$ and a walk W in G' from v to w such that $\mathsf{T}_W|\psi\rangle \propto |\phi\rangle$. Then, for any conflict-free CR induced on v with $|\psi\rangle$, w is assigned $|\phi\rangle$.*

Proof. We show that there is a walk $\tilde{W}$ in G' which is followed by the CR, for which $\mathsf{T}_{\tilde{W}}|\psi\rangle \propto |\phi\rangle$. Suppose $W = (v, u_\ell, \ldots, u_1, u_0)$ for $u_0 := w$. For each $i \geq 0$, let W_i denote the segment $(v, u_\ell, \ldots, u_i)$ of the walk W. Let m be the smallest integer such that the CR traverses W_m. If $m = 0$, then we may take $\tilde{W} = W$ is the walk followed by the CR from v to w. Otherwise, we show a reduction to "deform" W, to obtain walks W', W'', ..., and a decreasing sequence $m > m' > m'' > \cdots$, for which the CR follows the walks W_m, $W'_{m'}$, $W''_{m''}$, *etc.*. These walks have successively shorter "tails" of edges which are not followed by the CR: the final such walk $\tilde{W}$ is then one which is completely followed by the CR.

Given that $m > 0$, let $|\psi_m\rangle = \mathsf{T}_{W_m}|\psi\rangle$. By hypothesis, the CR does not traverse the edge (u_m, u_{m-1}), either because $\mathsf{T}_{u_m u_{m-1}}|\psi_m\rangle = 0$, or because of an assignment on u_{m-1}. The former implies $\mathsf{T}_W|\psi\rangle = 0 \not\propto |\phi\rangle$, contrary to hypothesis. Then there is a walk $W'_{m-1} = (v, u'_r, \cdots, u'_m, u_{m-1})$ in G', which is followed by the CR to make the assignment to u_{m-1}. (Note that the assignments to u_{m-1} made by both W and W'_{m-1} are proportional to one another, as otherwise the CR would have detected a conflict when attempting to traverse edge (u_m, u_{m-1}) during its breadth-first search.) We extend the walk W'_{m-1} to a walk $W' = (v, u'_r, \ldots, u'_m, u_{m-1}, \ldots, u_1, w)$. The CR has traversed W' at least as far as the vertex

u_{m-1}, missing out fewer edges at the end than it does for W. Furthermore, as the CR is conflict-free, we have $\mathsf{T}_{u_1 w}\mathsf{T}_{u_2 u_1}\cdots\mathsf{T}_{u_m u_{m-1}}|\psi_m\rangle \propto \mathsf{T}_{W'}|\psi\rangle$, so that $|\phi\rangle \propto \mathsf{T}_W|\psi\rangle \propto \mathsf{T}_{W'}|\psi\rangle$ by construction.

Repeating the reduction above yields a walk $\tilde{W}$ in G' which is completely followed by the CR, for which $\mathsf{T}_{\tilde{W}}|\psi\rangle \propto |\phi\rangle$ by induction. Then $|\phi\rangle$ is the assignment made to w by the CR. ◀

Note that the above result holds regardless of which walk W we consider from v to w, so long as $\mathsf{T}_W|\psi\rangle \neq 0$. Thus a conflict-free CR induced at v depends on a consistency between all walk operators, from v to any other given w, relative to the initial assignment $|\psi_v\rangle$. For the case $w = v$, we then have:

▶ **Lemma 3.4** (Circuit Lemma). *Let W be a closed walk starting and ending at v. If $|\psi_v\rangle$ is not an eigenvector of T_W, then inducing a CR at v with $|\psi_v\rangle$ yields a conflict.*

Proof. By definition, the CR assigns $|\psi_v\rangle$ to v. If the CR is conflict-free, then either $\mathsf{T}_W|\psi_v\rangle = 0$ or $\mathsf{T}_W|\psi_v\rangle \propto |\psi_v\rangle$, by Unique Assignment (Lemma 3.3). Thus, if $|\psi_v\rangle$ is not an eigenvector of T_W, such a CR will have a conflict. ◀

Lemma 3.3 also allows us to decouple the set of vertices given assignments by a CR, from the rest:

▶ **Lemma 3.5** (Unilateral Lemma). *For any state $|\psi\rangle$ and vertex v, suppose that a CR C_1 induced at v with $|\psi\rangle$ is conflict-free. Let A denote the set of vertices given an assignment by C_1, and $|\psi_a\rangle$ denote the assignment made by C at a given $a \in A$. Then, for any constraint Π_{ab} for $a \in A$ and $b \in V \setminus A$ and for any $|\phi\rangle \in \mathcal{H}$, $\Pi_{ab}\big(|\psi_a\rangle \otimes |\phi\rangle\big) = 0$.*

Proof. For $a \in A$, the CR C_1 must discover a walk $W = (v, v_1, v_2, \ldots, v_\ell)$ for $v_\ell := a$, such that for any sub-walk $W_i = (v, v_1, \ldots, v_i)$ for $1 \leq i \leq \ell$, we have $\mathsf{T}_{W_i}|\psi\rangle \neq 0$. The assignment made to a by C_1 is then $|\psi_a\rangle := \mathsf{T}_W|\psi\rangle$ by construction. Conversely, as $b \notin A$, it follows by the Unique Assignment (Lemma 3.3) that all walks W_* in G' from v to w satisfy $\mathsf{T}_{W_*}|\psi\rangle = 0$: this holds in particular for the walk $W' = (v, v_1, \ldots, a, b)$. Then $\mathsf{T}_{ab}|\psi_a\rangle = 0$, which is to say that $\Pi_{ab}\big(|\psi_a\rangle \otimes |\phi\rangle\big) = 0$ for all $|\phi\rangle$. ◀

The Unilateral Lemma allows us to treat conflict-free CRs as "set-and-forget" subroutines, in which we establish partial assignments on a set of qubits which we may remove from an instance $\mathcal{P} = \{\Pi_{ij}\}_{ij\in E}$ of 2-QSAT, obtaining a simpler, equivalent instance $\mathcal{P}' \subset \mathcal{P}$. Formally, we have the following.

▶ **Theorem 3.6** (Set-and-Forget Theorem). *Let $\mathcal{P} = \{\Pi_{ij}\}_{ij\in E}$ be an instance of 2-QSAT with interaction graph $G' = (V, E)$. Suppose that C is a conflict-free CR induced at $v \in V$ with $|\psi_v\rangle \in \mathcal{H}$, and let A denote the set of vertices given assignments by C. Let $\mathcal{P}'$ be a 2-QSAT instance obtained from $\mathcal{P}$ by removing all constraints acting on A. Then $\mathcal{P}$ is satisfiable by product states if and only if $\mathcal{P}'$ is.*

Proof. For a given $a \in A$, let $|\psi_a\rangle$ denote the assignment made by C to a. By construction, the states $|\psi_a\rangle$ jointly satisfy all constraints between vertices in a; and by the Unilateral Lemma (Lemma 3.5), the states $|\psi_a\rangle$ also unilaterally satisfy constraints between vertices in A and vertices in $V \setminus A$. If $\mathcal{P}'$ is satisfiable by a state $|\Phi\rangle = \bigotimes_{v\in V\setminus A}|\phi_a\rangle$, then $\mathcal{P}$ is satisfiable by $|\Psi\rangle = \big[\bigotimes_{a\in A}|\psi_a\rangle\big] \otimes |\Phi\rangle$. For the converse, suppose that $\mathcal{P}$ is satisfiable by some state $|\Psi'\rangle = \bigotimes_{v\in V}|\psi'_v\rangle$ (which may not agree with the assignments made by C). Define $|\Psi\rangle = \big[\bigotimes_{a\in A}|\psi_a\rangle\big] \otimes \big[\bigotimes_{v\in V\setminus A}|\psi'_v\rangle\big]$. Again, $|\Psi\rangle$ satisfies all constraints acting on vertices $a \in A$, and by construction it also satisfies all constraints internal to $V \setminus A$. Then $|\Psi\rangle$ also satisfies $\mathcal{P}$, and its restriction to $V \setminus A$ satisfies $\mathcal{P}'$. ◀

3.2 How to find conflict-free chain reactions efficiently

The Set-and-Forget Theorem (Theorem 3.6) provides us with the following approach to find a product assignment for an instance $\mathcal{P}$ of 2-QSAT: (*i*) pick an unassigned vertex v, (*ii*) find $|\psi_v\rangle$ such that the CR induced at v with $|\psi_v\rangle$ is conflict-free, and (*iii*) use this CR to produce a partial assignment, reducing to an instance $\mathcal{P}'$ with fewer qubits. It remains to attempt to find such a state $|\psi_v\rangle$, or determine that none exist, from the continuum $\mathcal{H}$ of single-qubit states.

As we describe in Section 1, and as shown by the Circuit Lemma (Lemma 3.4), it suffices for us to restrict our search for $|\psi\rangle$ to the eigenvectors of T_W for a closed walk W, *e.g.* a cycle. Define a *discretizing cycle* as a directed cycle C (starting and ending at some vertex v) with cycle matrix $\mathsf{T}_C \not\propto^* I$. For such cycles, the Circuit Lemma allows us to narrow down our search for $|\psi_v\rangle$ to the eigenvectors of T_C, of which there are at most two. This raises two questions: (1) How to find discretizing cycles efficiently, and (2) how to deal with variables which are not on any discretizing cycle.

As noted in Section 1, product operators complicate the task of detecting discretising cycles, but also provide a second way to narrow the search for assignments $|\psi_v\rangle$ leading to conflict-free CRs.

▶ **Lemma 3.7** (Product Constraint Lemma)**.** *In a product-satisfiable instance of 2-QSAT with a rank-1 product constraint projecting onto a state $|\phi_{uv}\rangle = |\gamma_u\rangle \otimes |\gamma_v\rangle$, at least one of the CRs at vertex u or v with states $|\gamma_u^\perp\rangle$ or $|\gamma_v^\perp\rangle$, respectively, is conflict-free.*

Proof. Suppose that the instance is product satisfiable, but that a CR starting at qubit u with state $|\gamma_u^\perp\rangle$ has a conflict. Then by the Conflict Lemma (Lemma 3.2), for any satisfying product state $|\psi\rangle = \bigotimes_{v\in V} |\psi_v\rangle$, we have $|\psi_u\rangle \not\propto |\gamma_u^\perp\rangle$. By construction, we have $|\psi_v\rangle \propto \mathsf{T}_{uv}\,|\psi_u\rangle = |\gamma_v^\perp\rangle \neq 0$. Thus a CR induced at v with $|\gamma_v^\perp\rangle$ will be conflict-free (as otherwise $|\psi\rangle$ cannot be a satisfying assignment, again by the Conflict Lemma). ◀

Using Lemma 3.7 together with the Set-And-Forget Theorem (Theorem 3.6), we may find a partial assignment satisfying any given product constraint; repeating this for all product constraints will either (*i*) reveal that the original 2-QSAT instance is unsatisfiable, (*ii*) yield a satisfying assignment for the entire instance, or (*iii*) yield an equivalent instance of 2-QSAT in which all constraints are projectors onto *entangled* states.

Let us call an instance of 2-QSAT *irreducible* if it has a connected interaction graph G', and all of its constraints are rank-1 projectors onto entangled states. In such an instance of 2-QSAT, all transfer matrices are invertible. A conflict-free CR induced at any vertex will yield assignments for every other vertex; thus, a single discretizing cycle suffices to determine whether or not the instance is satisfiable. We show that when a discretising cycle is present in such an instance of 2-QSAT, it is easily found:

▶ **Lemma 3.8.** *Suppose G' is an interaction graph of an irreducible instance of 2-QSAT, which contains a discretizing cycle C. Let $T \subset G'$ be a tree which contains all of the vertices of C. Then there is at least one edge e in C, such that the (unique) cycle in the graph $T \cup \{e\}$ is a discretizing cycle.*

Proof. In the tree T, there is a unique path P_{vw} from any given vertex $v \in V$ to any other connected vertex w. Furthermore, by the irreducibility of the 2-QSAT instance, $\mathsf{T}_{P_{vw}}$ is non-singular in each case. Suppose that $C = (v_1, v_2, \ldots, v_\ell, v_1)$ is a discretizing cycle in the implication graph G. Consider the closed walk from v_1 to itself in T, given by $W = P_{v_1v_2}P_{v_2v_3}\cdots P_{v_\ell v_1}$. By induction, we may show that the truncated walk $W' =$

$P_{v_1v_2}P_{v_2v_3}\cdots P_{v_{\ell-1}v_\ell}$ satisfies $\mathsf{T}_{W'} \propto \mathsf{T}_{P_{v_1v_\ell}} \propto \mathsf{T}_{P_{v_\ell v_1}}^{-1}$ for each ℓ: thus $\mathsf{T}_W \propto I$. However, $\mathsf{T}_C = \mathsf{T}_{v_\ell v_1}\cdots\mathsf{T}_{v_2v_3}\mathsf{T}_{v_1v_2} \not\propto I$ by hypothesis. Then there is an edge vw in C for which $\mathsf{T}_{vw} \not\propto \mathsf{T}_{P_{vw}}$. Then the unique cycle C' in $T \cup \{vw\}$ contains the path P_{vw} from v to w, as well as the edge vw, and has cycle matrix $\mathsf{T}_{C'} = \mathsf{T}_{wv}\mathsf{T}_{P_{vw}} \propto \mathsf{T}_{vw}^{-1}\mathsf{T}_{P_{vw}} \not\propto I$. ◀

▶ **Theorem 3.9** (Cycle Discovery Theorem). *Suppose G' is the interaction graph of an irreducible instance of 2-QSAT, and contains a discretizing cycle C. Then a depth-first search from any vertex $v \in V$, in which each edge is traversed at most once, suffices to discover a discretizing cycle C'.*

Proof. Consider a DFS starting from any vertex $v \in V$. Define a tree $T \subset G$, in which each edge e traversed by the DFS is included if and only if e is traversed for the first time some vertex is visited. As the DFS reaches each vertex w, it also computes the path operator $\mathsf{T}_{P_{vw}}$ for the path taken from v to w. Each time the DFS traverses an edge uw from some vertex u to a vertex w which it has previously visited, it tests whether $\mathsf{T}_{P_{vu}} \propto \mathsf{T}_{wu}\mathsf{T}_{P_{vw}}$. If so, it continues the DFS from w. Otherwise the cycle C' consisting of $P^{\mathsf{R}}_{vu}P_{vw}$ followed by wu is discretizing, as $\mathsf{T}_{C'} \propto \mathsf{T}_{uw}\mathsf{T}_{P_{vu}}\mathsf{T}_{P_{vw}}^{-1} \not\propto^* I$. Conversely by Lemma 3.8, if G has a discretizing cycle, the DFS must eventually traverse such an edge. ◀

Implicit in Theorem 3.9 is a linear-time algorithm for finding discretising cycles in an irreducible instance of 2-QSAT, when one is present. It remains to describe how to treat irreducible instances which have no discretizing cycles. The absence of any means of discretising the state-space of any qubit in such an instance actually represents freedom of choice in this case; while this is implicit in Refs. [3, 21, 9], we prove it here for the sake of completeness.

▶ **Lemma 3.10** (Free Choice Lemma). *In an irreducible instance of 2-QSAT with no discretizing cycles, any choice of single-qubit state $|\psi_v\rangle$ for some v in the component gives rise to a conflict-free CR.*

Proof. Let G' be the interaction graph. Consider a CR induced at v with $|\psi_v\rangle$, and consider the paths P_{vw} to each vertex w, by which the CR makes its first assignment $|\psi_w\rangle := \mathsf{T}_{P_{vw}}|\psi_v\rangle$ to w. If P'_{vw} is another walk from v to w, we have $\mathsf{T}_{P^{\mathsf{R}}_{vw}}\mathsf{T}_{P'_{vw}} \propto I$, from the hypothesis that there are no discretising cycles; then $\mathsf{T}_{P'_{vw}} \propto \mathsf{T}_{P_{vw}}$. Thus, regardless of the choice of $|\psi_v\rangle$, a consistent assignment $\mathsf{T}_{P'_{vw}}|\psi_v\rangle$ is computed every time the CR traverses an edge to visit w. ◀

4 A linear-time 2-QSAT algorithm

We finally present our 2-QSAT algorithm in Figure 1, whose correctness follows immediately by combining the results of Section 3. Following [10], we implement CRs (corresponding to their trial assignments) in parallel to ensure a linear bound on run-time; this is expanded upon in Section 5.

Preprocessing stage to impose input constraints

For conciseness, we present SOLVE$_{\mathrm{Q}}$ in Figure 1 with restrictions on the inputs it takes. As we indicate in Section 2, following Chen *et al.* [4], these restrictions ensure that the instance presented to SOLVE$_{\mathrm{Q}}$ is either satisfiable by a product state or unsatisfiable. These restrictions can be imposed through a pre-processing phase, as follows. For each pair $\{u, v\}$ subject to multiple constraints, sum the projectors to obtain positive semidefinite operator S_{uv}. Then perform the following:

Input: An instance of 2-QSAT consisting of rank-1 projectors $\mathcal{P} = \{\Pi_{ij}\}$ with interaction graph $G' = (V, E)$, with at most two parallel edges (u, v) per distinct $\{u, v\} \subset V$.

1. *Discretize on product constraints* – While there exists a projector $\Pi_{ij} = |\phi_{ij}\rangle\langle\phi_{ij}|$ such that $|\phi_{ij}\rangle = |\gamma_i\rangle \otimes |\gamma_j\rangle$ is a product state: simulate CRs at each $v \in \{i, j\}$ with $|\gamma_v^{\perp}\rangle$, in parallel.
   a. If conflicts arise in both CRs, halt and output "UNSAT".
   b. Fix the assignments for the first conflict-free CR that terminates, remove the set A of vertices that it visited from G', and go to Step 1.
2. *Discretize on cycles* – While there exists $v \in V$: search for a discretizing cycle $C \subseteq G'$ in the same connected component of v.
   - If such a cycle C is found at a vertex u: Let T_C be its cycle matrix, and S denote the set of eigenvectors of T_C. Simulate CRs at u with each $|\psi_u\rangle \in S$, in parallel.
     a. If conflicts arise in both CRs, halt and output "UNSAT".
     b. Fix the assignments for the first conflict-free CR that terminates, remove the set A of vertices that it visited from G', and go to Step 2.
   - If no such cycle is found: Induce a CR at v with $|\psi_v\rangle := |0\rangle$. Fix assignments made by the CR, remove the set A of vertices that it visits from G', and go to Step 2.
3. *Normalize* – For each qubit v, compute whether the assignment $|\psi_v\rangle$ is normalised: if not, compute a normalised version $|\psi_v\rangle := |\psi_v\rangle \,/\, \sqrt{\langle\psi_v|\psi_v\rangle}$.

Output: "UNSAT", or unit vectors $|\psi_v\rangle \in \mathcal{H}$ for each $v \in V$ which jointly satisfy $\mathcal{P}$.

Figure 1 An algorithm for 2-QSAT, denoted SOLVE_{Q}.

1. If any pair $\{u, v\}$ has $\text{rank}(S_{uv}) = 4$, halt with output UNSAT (as $\ker(S_{uv})$ contains no states).
2. For each pair $\{u, v\}$ with $\text{rank}(S_{uv}) = 2$, replace the constraints on $\{u, v\}$ with $\Pi_{uv,1} = |\eta_1\rangle\langle\eta_1|$ and $\Pi_{uv,2} = |\eta_2\rangle\langle\eta_2|$, for linearly independent columns $|\eta_1\rangle, |\eta_2\rangle$ of S_{uv}.
3. For each pair $\{u, v\}$ with $\text{rank}(S_{uv}) = 3$, record the unique state $|\psi_{uv}\rangle$ which spans $\ker(S_{uv})$ as a joint assignment to (u, v), and remove the constraints on $\{u, v\}$. If $|\psi_{uv}\rangle = |\psi_u\rangle \otimes |\psi_v\rangle$, record $|\psi_u\rangle$ and $|\psi_v\rangle$ as assignments to u and v respectively. (If any qubit is subject to conflicting assignments, halt with output UNSAT.)
4. For each pair $\{u, v\}$ given an assignment $|\psi_{uv}\rangle$ in the preceding step:
   - If $|\psi_{uv}\rangle = |\psi_u\rangle \otimes |\psi_v\rangle$, induce CRs (sequentially) at u with $|\psi_u\rangle$ and at v with $|\psi_v\rangle$.
   - If not, and there are non-product constraints Π_{iu} or Π_{iv} for any i, halt with output UNSAT (as any state of i is compatible only with product states on $\{u, v\}$). Otherwise, for each $\Pi_{iu} = |\gamma_i\rangle\langle\gamma_i| \otimes |\gamma_u\rangle\langle\gamma_u|$ or $\Pi_{iv} = |\gamma_i\rangle\langle\gamma_i| \otimes |\gamma_v\rangle\langle\gamma_v|$, induce a CR (sequentially) at i with $|\gamma_i^{\perp}\rangle$.

   For any CR induced, halt (with output UNSAT) either if the CR has a conflict, or if it makes an assignment to some other qubit w which has been given a different assignment as a result of a rank-3 constraint. If no conflict is detected, record the assignments, and remove the set A of qubits given assignments from G'.

This preprocessing phase involves much the same subroutines as SOLVE_{Q} itself, and does not contribute to the asymptotic run-time. (We include these steps in our detailed runtime analysis in the full version.)

5 Runtime analysis

We briefly sketch the runtime analyses for $\mathrm{SOLVE}_{\mathrm{Q}}$ in terms of field operations over $\mathbb{C}$ and bit operations, and discuss an optimization for the setting of product state constraints. A more in-depth treatment is given in the full version. We assume a random-access machine, so that memory access takes unit time. The constraints Π_i are specified as 4×4 matrices with coefficients from a finite-degree field extension $\mathbb{F}:\mathbb{Q}$, whose specification is also part of the input; arithmetic operations over such number fields can be performed efficiently [5]. From this representation we extract the basis vectors $|\eta_i\rangle$ for the image of Π_i by taking columns of Π_i, and omit normalisation: $\mathrm{SOLVE}_{\mathrm{Q}}$ then uses $|\eta_i\rangle$ to represent Π_i. Vectors are only normalised as the final step of the algorithm.

Field operations

$\mathrm{SOLVE}_{\mathrm{Q}}$ requires $O(n+m)$ operations over $\mathbb{C}$, for n and m the number of variables and clauses, respectively. As each vector $|\eta_i\rangle$ is in $\mathbb{C}^4$, operations on them (such as determining if $|\eta_i\rangle$ is a product constraint in Step 1) require $O(1)$ field operations. Following EIS [10], we simulate CRs in parallel by interleaving their steps, terminating both simulations as soon as one of them is found to be conflict-free. In the preprocessing phase and in Step 1b, this ensures that the number of vertices and edges removed (upon completion of a conflict-free CR) is proportional to the number of vertices and edges visited during the parallel CRs. Hence, the total number of edge-traversals of $\mathrm{SOLVE}_{\mathrm{Q}}$ is $O(m)$. Finally, by Step 2, the instance has been simplified to a disjoint union of irreducible instances. Theorem 3.9 ensures that if a discretizing cycle exists in any of the components, it can be found by a depth-first search; moreover, a single conflict-free CR suffices to assign satisfying states to all vertices in each component.

Bit operations

The bit-complexity of $\mathrm{SOLVE}_{\mathrm{Q}}$ differs from the field-operation complexity, for the simple reason that multiplying k transfer matrices yields a path matrix with $O(k)$-bit entries. Thus, operations such as determining the eigenvectors of such matrices, or whether $|\psi\rangle \propto |\phi\rangle$ for vectors in the image of these matrices, can take time $O(M(k))$, where $M(k)$ is the time to multiply two k-bit integers. This follows from the fact that computing $\sqrt{D} \in \mathbb{Z}$ for a perfect square $D \in \mathbb{Z}$ can be performed in $O(M(n))$ time using Newton's method (see *e.g.* Theorem 9.28 of Ref. [26]); and that equality testing over $\mathbb{Q}$ is bounded by $O(M(n))$, for rationals $r, s \in \mathbb{Q}$ with n bit representations as ratios. (To test whether $\frac{a}{b}$ and $\frac{c}{d}$ are equal, one tests whether $ad - bc = 0$.) Since the number of times we might need to compute eigenvectors or decide proportionality may scale as $m+n$, the runtime of $O((m+n)M(n))$ follows.

It may be necessary for $\mathrm{SOLVE}_{\mathrm{Q}}$ to represent its output using further field extensions $\mathbb{E}:\mathbb{F}$, for instance, when solving the characteristic polynomial $\det(\lambda I - \mathsf{T}_C)$ of a cycle matrix T_C, if the discriminant $D = (\operatorname{Tr} \mathsf{T}_C)^2 + 4(\det \mathsf{T}_C)$ is not a perfect square in $\mathbb{F}$. However, by the Set and Forget Theorem 3.6, any extension required by a CR will be independent of the CRs involved in the assignments made by other CRs; furthermore, the extensions involved in each CR is only quadratic, and specifically by a square root $\sqrt{D}$ of an element $D \in \mathbb{F}$.

The approach taken to the quadratic extensions by $\mathrm{SOLVE}_{\mathrm{Q}}$ is unconventional. Specifically, unless $D \in \mathbb{Q}$, we do not evaluate whether or not $\sqrt{D}$ is in $\mathbb{F}$ before defining the (possibly trivial) "extension" $\mathbb{E} = \mathbb{F}[\sqrt{D}]$. That is, we allow representations of number fields

$\mathbb{F}_k := \mathbb{Q}[\omega_1, \omega_2, \ldots, \omega_k]$ in which $\omega_j = \sqrt{\alpha_j}$ for some $\alpha_j \in \mathbb{F}_{j-1}$ (possibly including the case $\omega_1 = \sqrt{s}$ for $s \in \mathbb{Q}$), and where it may come to pass that $\omega_j \in \mathbb{F}_{j-1}$. This prevents us from easily presenting coefficients in a normal form: crucially however, it is still possible for us to perform equality tests and arithmetic operations in time $O(M(n))$, for $\alpha \in \mathbb{F}_k$ expressed as $\frac{1}{\mu} f(\omega_1, \ldots, \omega_k)$ for $\mu \in \mathbb{Z}$ and $f \in \mathbb{Z}[x]$ with coefficients of size $O(n)$, provided that k is bounded by a constant. (In the case of $\mathrm{SOLVE}_\mathrm{Q}$, we bound $k \leq 3$.)

Thus while the output of $\mathrm{SOLVE}_\mathrm{Q}$ may not be reduced, it nevertheless presents exact, normalised, satisfying states by means of tensor factors. Complete details are to be found in the full version.

Reduced complexity of 2-QSAT for product constraints

Using a simple optimization which exploits product constraints, $\mathrm{SOLVE}_\mathrm{Q}$ can in fact accept inputs over any field extension $\mathbb{F}{:}\mathbb{Q}$ (algebraic or otherwise), and solve them with $O(n+m)$ bit operations provided that all projectors are product operators. This requires only that arithmetic operations and equality testing against 0 can be performed in $\mathbb{F}$ in $O(1)$ time on inputs with representations of size $O(1)$. Specifically: the transfer matrix of a product constraint $\Pi_{uv} = |\gamma_u\rangle\langle\gamma_u| \otimes |\gamma_v\rangle\langle\gamma_v|$ is $\mathsf{T}_{uv} \propto |\gamma_v^\perp\rangle\langle\gamma_u|$, whose image is spanned by $|\gamma_v^\perp\rangle$. For any assignment $|\psi_u\rangle$ to u, if $\mathsf{T}_{u,v}|\psi_u\rangle \neq 0$, we can set v to $|\gamma_v^\perp\rangle$ (which by assumption on the input requires $O(1)$ bits), as opposed to the potentially more complex vector $\mathsf{T}_{u,v}|\psi_u\rangle \propto |\gamma_v^\perp\rangle$. Thus, in Step 1, the complexity of the assignments made by a CR are no more complex than the vectors of the projectors Π_{uv} in the input, so that all algebraic operations may be performed in $\Theta(1)$ time rather than $O(M(n))$ time. In particular, for classical 2-SAT instances, we recover an $O(m+n)$ upper bound on the bit-complexity of $\mathrm{SOLVE}_\mathrm{Q}$, matching the asymptotic performance of the APT and EIS algorithms [2, 10].

6 On lower bounds for bit complexity

Most investigations into 2-QSAT are presented in terms of unit-cost operations over some algebraic number field $\mathbb{F}$. As a result, no restrictions are usually put on how the output of a classical solution to 2-QSAT is represented. To consider lower bounds on the bit-complexity of presenting a solution to 2-QSAT, it becomes necessary to consider what restrictions to impose on the output, as without such restrictions the notion of what form the output may take becomes ill-defined. We impose the restriction of outputs which are *rationalised*, as follows. Let $\mathbb{F} = \mathbb{Q}[\omega]$ be an algebraic number field, so that ω is an algebraic number whose minimal polynomial p is a monic polynomial over $\mathbb{Z}$. An element $\alpha \in \mathbb{F}$ is presented in *rationalised form* by an expression of the form $f(\omega)/D = \alpha$, where $D > 0$ is an integer and $f \in \mathbb{Z}[x]$ is an polynomial such that $\deg(f) < \deg(p)$. Despite the unconventional representation described in Section 5, this is one constraint which the output of $\mathrm{SOLVE}_\mathrm{Q}$ respects.

There are further restrictions which one might consider, such as the output state vectors being normalised (which $\mathrm{SOLVE}_\mathrm{Q}$ satisfies), and that they be *reduced*: that the coefficients $\alpha = f(\omega)/D$ satisfy $\gcd(f, D) = 1$. Consider, for instance, an algorithm which produces its output in *minimal form*: each state that it outputs is normalised, in reduced rationalised form, and involves the minimal field extension $\mathbb{F}{:}\mathbb{Q}$ necessary to do so, represented as $\mathbb{F} = \mathbb{Q}[\omega]$ where the minimal polynomial of ω is a monic polynomial over the integers. While $\mathrm{SOLVE}_\mathrm{Q}$ does not compute outputs in minimal form (*e.g.*, it may fail to produce outputs in reduced form), we show that the multiplication time $O(M(n))$ for n bit integers is a relevant lower

bound for algorithms which do, suggesting that the role of $M(n)$ in the upper bound of $\mathrm{SOLVE}_{\mathrm{Q}}$ is not merely accidental.

▶ **Lemma 6.1.** *There exist instances of 2-QSAT on n vertices and $m \in O(n)$ clauses, such that exhibiting a requested tensor factor of a satisfying solution, in minimal form, requires $\Omega(M(n))$ bit operations in in the worst case.*

Proof. Let M and N be positive, odd n-bit integers, with binary expansions $M = \sum_{t=0}^{n-1} 2^t M_t$ and $N = \sum_{t=0}^{n-1} 2^t N_t$, where $M_i, N_i \in \{0,1\}$ for each $0 \leq i < n$. We construct an instance of 2-QSAT whose unique product state solution is one in which one of the qubits q is assigned a state

$$|\psi_q\rangle \;:=\; \frac{M}{\sqrt{D}}\,|0\rangle + \frac{2^n + MN}{\sqrt{D}}\,|1\rangle \;=\; \frac{M\sqrt{D}}{D}\,|0\rangle + \frac{(2^n + MN)\sqrt{D}}{D}\,|1\rangle\,, \tag{3}$$

where $D = M^2 + (2^n + MN)^2$. Either the middle or the right-hand expression in Eqn. (3) is in rationalised and normalised form, depending on whether D is a perfect square. As M, $2^n + MN$, and D are coprime, that rationalised expression is in reduced form, if $\mathbb{F} = \mathbb{Q}[\sqrt{D}]$. If D is neither a perfect square nor square-free, it may be that $\sqrt{D}$ is represented as $\delta\sqrt{D'} \in \mathbb{Q}[\sqrt{D'}]$, where $D = D'\delta^2$. In this case, by hypothesis, a representation of $|\psi_q\rangle$ in reduced form would be identical (up to signs) to

$$|\psi_q\rangle = \frac{M\sqrt{D'}}{D'\delta}\,|0\rangle + \frac{(2^n + MN)\sqrt{D'}}{D'\delta}\,|1\rangle\,. \tag{4}$$

In any case, the minimal form representation would provide a specification of the extension element $\sqrt{D'}$, the denominators $D'\delta$, and the numerators $A = M$ and $B = 2^n + MN$ (or $A = -M$ and $B = -2^n - MN$, which yields an equivalent vector in $\mathbb{Q}[\sqrt{D'}]$). From such a representation, one could compute MN simply as $B - 2^n$ (or $-B - 2^n$ respectively), which requires time $O(n)$.

The instance we construct is on a chain of $2n+2$ qubits, labelled $v \in \{0, 1, 2, \ldots, 2n{+}1\}$, as follows. For $1 \leq i \leq n$, we define matrices

$$\mathsf{T}_{i-1,i} = \begin{pmatrix} 1 & 0 \\ M_{n-i} & 2 \end{pmatrix}, \qquad \mathsf{T}_{n+i,n+1+i} = \begin{pmatrix} 1 & 0 \\ N_{n-i} & 2 \end{pmatrix}; \tag{5}$$

and also two matrices $\mathsf{T}_{n,n+1}$ and $\mathsf{T}'_{0,1}$:

$$\mathsf{T}_{n,n+1} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \qquad \mathsf{T}'_{0,1} = \begin{pmatrix} 0 & 1 \\ 0 & M_{n-1} \end{pmatrix}. \tag{6}$$

For each $i \in \{0, 1, 2, \ldots, 2n\}$, we include a constraint $\Pi_{i,i+1}$ between qubits i and $i+1$, with transfer matrix $\mathsf{T}_{i,i+1}$; and we also include a second constraint $\Pi'_{0,1}$ between 0 and 1, with transfer matrix $\mathsf{T}'_{0,1}$. The resulting instance of 2-QSAT has two rank-1 constraints between qubits 0 and 1, and one rank-1 constraint between all other consecutive pairs of qubits. By Chen *et al.* [4], this instance is then satisfiable by a product state if it is satisfiable at all. It is easy to show that all of the projectors have rational coefficients in this case, so we take the field of the representation to be $\mathbb{Q}$ itself.

We show that there is a unique product state which satisfies the above instance of 2-QSAT. It is easy to show that the opposite transfer operator to $\mathsf{T}'_{0,1}$ is

$$\mathsf{T}'_{1,0} \propto \begin{pmatrix} -M_{n-1} & 1 \\ 0 & 0 \end{pmatrix} \tag{7}$$

so that $\mathsf{T}'_{1,0}\mathsf{T}_{0,1} \propto |0\rangle\langle 1|$. The only eigenvector of this operator is $|0\rangle$, which is therefore the only single-qubit state on qubit 0 which is consistent with a satisfying solution. As all other transfer operators are non-singular, this determines a unique assignment for all other qubits i in the chain, determined by the first column of the walk operator $\mathsf{T}_{[0,i]} := \mathsf{T}_{i-1,i} \cdots \mathsf{T}_{1,2}\mathsf{T}_{0,1}$. It is easy to show for $1 \leq i \leq n$ that

$$\mathsf{T}_{[0,i]} = \begin{pmatrix} 1 & 0 \\ \sum\limits_{1 \leq t \leq i} M_{n-t} 2^{i-t} & 2^i \end{pmatrix}, \tag{8}$$

and that in particular

$$\mathsf{T}_{[0,n]} = \begin{pmatrix} 1 & 0 \\ M & 2^n \end{pmatrix}; \tag{9}$$

from this we easily obtain

$$\mathsf{T}_{[0,n+1]} = \begin{pmatrix} M & 2^n \\ 1 & 0 \end{pmatrix}; \tag{10}$$

from which point we may prove by induction for $1 \leq i \leq n$ that

$$\mathsf{T}_{[0,n+1+i]} = \begin{pmatrix} M & 2^n \\ 2^i + M \sum\limits_{1 \leq t \leq i} N_{n-t} 2^{i-t} & 2^n \sum\limits_{1 \leq t \leq i} N_{n-t} 2^{i-t} \end{pmatrix}; \tag{11}$$

so that

$$\mathsf{T}_{[0,2n+1]} = \begin{pmatrix} M & 2^n \\ 2^n + MN & 2^n N \end{pmatrix}. \tag{12}$$

Let q be qubit $2n+1$. The only assignment to this qubit which is consistent with a satisfying assignment is then the state given by the first column of $\mathsf{T}_{[0,2n+1]}$, which is $M|0\rangle + (2^n + MN)|1\rangle$; the vector given by Eqn. (3) is the normalised version of this vector.

Using the techniques of Laumann *et al.* [21], we may show that the space of satisfying assignments of this instance has dimension 2, spanned by the product solution above, and an entangled solution on all of the qubits. Considering all projectors except for $\Pi'_{0,1}$, there is an invertible (non-unitary) local transformation mapping all projectors $\Pi_{i-1,i}$ to $|\Psi^-\rangle\langle\Psi^-|$, the two-qubit antisymmetric projector. Thus the satisfying states for these projectors are the symmetric subspace on $S = 2n+2$ qubits, which is spanned by any collection of states of the form $|\alpha_i\rangle^{\otimes 2n+2}$, for $S+1 = 2n+3$ distinct states $|\alpha_i\rangle$. Any state in this space which is not a product state, is entangled across the entire chain of qubits. Undoing this change of local co-ordinates, it follows that any state which satisfies the above instance of 2-QSAT which is not a product state, is also entangled across the entire chain of qubits (*i.e.*, it cannot be factorized across any cut). Since we require each factor to be explicitly written in the standard basis, such a solution would then require explicitly writing out the standard basis elements of a vector of dimension 2^{2n+2}; such solutions would require vectors of dimension 2^{2n+2} to represent. Any algorithm which in polynomial time exhibits one of the tensor factors of the solution, must therefore exhibit factors of the product solution. In particular, it must compute $|\psi_q\rangle$ if this is the required tensor factor. As we have already shown an $O(n)$ reduction from computing the product MN to computing the minimal representation of $|\psi_q\rangle$, it follows that there is an $\Omega(M(n))$ lower bound for such an algorithm in the worst case. ◀

▶ **Corollary 6.2.** *If there does not exist a $\Theta(n)$-time algorithm for multiplying two n-bit integers, then there does not exist an $O(m+n)$-time algorithm to present single-qubit marginals of satisfying solutions to instances of 2-QSAT.*

We would also like to show lower bounds for algorithms such as SOLVE_{Q}, which do not necessarily compute its output in reduced form, but which does compute an *explicit* output, in the sense of presenting a complete description of a satisfying solution via tensor factors. We may obtain such lower bounds even for algorithms which produce non-normalised outputs, as follows.

▶ **Lemma 6.3.** *There exist instances of 2-QSAT on n vertices and $m \in O(n)$ clauses, such that an explicit rationalised (but not necessarily normalised) assignment for a satisfying state requires $\Omega(n^2)$ bits.*

Proof. We may simplify the construction of Lemma 6.1 by omitting the qubits $n+1, \ldots, 2n+1$ and the projectors which act on them. This yields an instance in which there is a unique product solution (with all other solutions requiring a vector of dimension 2^{n+1} to represent). In this product state, the qubit n is in a state $|\psi_n\rangle \propto |0\rangle + M\,|1\rangle$. More generally, each qubit $1 \leq i \leq n$ is in a state

$$|\psi_i\rangle \propto |0\rangle + M^{(i)}\,|1\rangle \tag{13}$$

where $M^{(i)} = \sum_{t=1}^{i} M_{n-t} 2^{i-t}$. As $M_{n-1}M_{n-2}\cdots M_2 M_1 \in \{0,1\}^{n-1}$ may be an arbitrary $n-1$ bit string, and as we require the tensor factors on the qubits i to be presented independently of one another, the integers $M^{(i)}$ cannot be represented any more succinctly in the worst case; at best, by applying arbitrary scalar factors, we may consider representations $|\psi_i\rangle = \frac{1}{\alpha_i}|0\rangle + \frac{M^{(i)}}{\alpha_i}|1\rangle$, in which the representation of the $|1\rangle$ component of $|\psi_i\rangle$ may be reduced if α_i divides $M^{(i)}$, but at the cost of increasing the size of the representation of the $|0\rangle$ component. (More formally, if the pair $(1/\alpha_i, M^{(i)}/\alpha_i)$ has asymptotically smaller Kolmogorov complexity than the pair $(1, M^{(i)})$, we would have a contradiction, since the former allows us to extract $M^{(i)}$ – thus, we would have a shorter description of $M^{(i)}$ than its Kolmogorov complexity allows.) Thus, for any constant $0 < \alpha < 1$, the qubits $\lfloor \alpha n \rfloor < i < n$ all require $\Omega(n)$ bits to represent, yielding a total lower bound of $\Omega(n^2)$. ◀

▶ **Corollary 6.4.** *Up to $\Omega(\log(n)^{1+o(1)})$ factors, SOLVE_{Q} is optimal among algorithms which present explicit expressions for satisfying assignments.*

Acknowledgements. NdB thanks Rick, Linda, Colin, and Michelle de Beaudrap for their hospitality during an academic absence, during which the bulk of this work was performed. This project was initiated while NdB was affiliated with the Centrum Wiskunde & Informatica, with support from the European Commission FET-Proactive project Quantum Algorithms (QALGO) 600700, and completed with support from the UK Quantum Technology Hub project NQIT. SG thanks Ronald de Wolf and Centrum Wiskunde & Informatica, where this work was initiated, for their hospitality. SG acknowledges support from a Government of Canada NSERC Banting Postdoctoral Fellowship, the Simons Institute for the Theory of Computing at UC Berkeley, and NSF grant CCF-1526189.

References

1 I. Arad, M. Santha, A. Sundaram, and S. Zhang. Linear time algorithm for quantum 2SAT. arXiv:1508.06340, 2015.

2 B. Aspvall, M. F. Plass, and R. E. Tarjan. A linear-time algorithm for testing the truth of certain quantified boolean formulas. *Information Processing Letters*, 8(3):121–123, 1979.

3 S. Bravyi. Efficient algorithm for a quantum analogue of 2-SAT. Available at arXiv.org e-Print quant-ph/0602108v1, 2006.

4 J. Chen, X. Chen, R. Duan, Z. Ji, and B. Zeng. No-go theorem for one-way quantum computing on naturally occurring two-level systems. *Physical Review A*, 83:050301(R), 2011.

5 H. Cohen. *A Course in Computational Algebraic Number Theory*. Graduate Texts in Mathematics. Springer, 1993.

6 S. Cook. The complexity of theorem proving procedures. In *Proceedings of the 3rd ACM Symposium on Theory of Computing (STOC 1972)*, pages 151–158, 1972.

7 M. Davis and H. Putnam. A computing procedure for quantification theory. *Journal of the ACM*, 7(3):201, 1960.

8 N. de Beaudrap. Difficult instances of the counting problem for 2-quantum-sat are very atypical. In *Proceeedings of TQC'14*, pages 118–140, 2014. arXiv:1403.1588.

9 N. de Beaudrap, T. J. Osborne, and J. Eisert. Ground states of unfrustrated spin hamiltonians satisfy an area law. *New Journal of Physics*, 12, 2010. arXiv:1009.3051.

10 S. Even, A. Itai, and A. Shamir. On the complexity of the time table and multi-commodity flow problems. *SIAM Journal on Computing*, 5(4):691–703, 1976.

11 M. Fürer. Faster integer multiplication. In *Proceedings of the 39th ACM Symposium on the Theory of Computing (STOC 2007)*, pages 55–67, 2007.

12 Oded Goldreich and David Zuckerman. Another proof that $\mathcal{BPP} \subseteq \mathcal{PH}$ (and more). In Oded Goldreich, editor, *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation*, volume 6650 of *Lecture Notes in Computer Science*, pages 40–53. Springer Berlin Heidelberg, 2011. doi:10.1007/978-3-642-22670-0_6.

13 D. Gosset and D. Nagaj. Quantum 3-SAT is QMA1-complete. In *Proceedings of the 54th IEEE Symposium on Foundations of Computer Science (FOCS 2013)*, pages 756–765, 2013.

14 Z. Ji, Z. Wei, , and B. Zeng. Complete characterization of the ground space structure of two-body frustration-free hamiltonians for qubits. *Physical Review A*, 84, 2011.

15 S. P. Jordan, H. Kobayashi, D. Nagaj, and H. Nishimura. Achieving perfect completeness in classical-witness quantum Merlin-Arthur proof systems. *Quantum Information & Computation*, 12(5 & 6):461–471, 2012.

16 R. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, pages 85–103. New York: Plenum, 1972.

17 J. Kempe, A. Kitaev, and O. Regev. The complexity of the local Hamiltonian problem. *SIAM Journal on Computing*, 35(5):1070–1097, 2006.

18 A. Kitaev. Fault-tolerant quantum computation by anyons. *Annals of Physics*, 303(1):2–30, 2003.

19 A. Kitaev, A. Shen, and M. Vyalyi. *Classical and Quantum Computation*. American Mathematical Society, 2002.

20 M. R. Krom. The decision problem for a class of first-order formulas in which all disjunctions are binary. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik*, 13:15–20, 1967.

21 C. R. Laumann, R. Moessner, A. Scardicchio, and S. L. Sondhi. Phase transitions and random quantum satisfiability. *Quantum Information & Computation*, 10:1–15, 2010.

22 L. Levin. Universal search problems. *Problems of Information Transmission*, 9(3):265–266, 1973.

23 C. Papadimitriou. On selecting a satisfying truth assignment. In *Proceedings of the 32nd Annual IEEE Symposium on Foundations of Computing (FOCS 1991)*, pages 163–169, 1991.

24 W. V. Quine. On cores and prime implicants of truth functions. *The American Mathematical Monthly*, 66(5):755–760, 1959.

25 R. E. Tarjan. Depth fist search and linear graph algorithms. *SIAM Journal on Computing*, pages 146–160, 1972.

26 J. von zur Gathen and J. Gerhard. *Modern Computer Algebra*. Cambridge University Press, 2003.

27 S. Zachos and M. Furer. Probabalistic quantifiers vs. distrustful adversaries. In *Foundations of Software Technology and Theoretical Computer Science, 7th Conference*, pages 443–455, 1987. Volume 287 of *Lecture Notes in Computer Science*.

Complexity Classification of Two-Qubit Commuting Hamiltonians

Adam Bouland[1], **Laura Mančinska**[2], **and Xue Zhang**[3]

1 **Massachusetts Institute of Technology, Cambridge, USA**
adam@csail.mit.edu
2 **Centre for Quantum Technologies, National University of Singapore, Singapore**
laura@locc.la
3 **Massachusetts Institute of Technology, Cambridge, USA**
lzh@mit.edu

Abstract

We classify two-qubit commuting Hamiltonians in terms of their computational complexity. Suppose one has a two-qubit commuting Hamiltonian H which one can apply to any pair of qubits, starting in a computational basis state. We prove a dichotomy theorem: either this model is efficiently classically simulable or it allows one to sample from probability distributions which cannot be sampled from classically unless the polynomial hierarchy collapses. Furthermore, the only simulable Hamiltonians are those which fail to generate entanglement. This shows that *generic* two-qubit commuting Hamiltonians can be used to perform computational tasks which are intractable for classical computers under plausible assumptions. Our proof makes use of new postselection gadgets and Lie theory.

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

Keywords and phrases Quantum Computing, Sampling Problems, Commuting Hamiltonians, IQP, Gate Classification Theorems

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.28

1 Introduction

Quantum computers hold the promise of performing computational tasks which cannot be simulated efficiently using classical computers. A hallmark example of this is Shor's quantum factoring algorithm [33] for which no classical analog is known. However, proving that quantum computers hold an advantage over classical ones when it comes to factoring or any other decision problem would show that $\mathsf{P} \neq \mathsf{PSPACE}$, which is well beyond our current reach. Therefore, we aim to establish quantum advantage under widely accepted complexity assumptions like $\mathsf{P} \neq \mathsf{NP}$, non-collapse of the polynomial hierarchy PH, and others. In this submission we show that generic two-qubit commuting Hamiltonians can be used to perform computational tasks which are intractable for classical computers unless PH collapses. Since commuting gate sets allow for easier fault-tolerant implementation [5], our results offer the possibility to experimentally perform classically intractable computations even before achieving universal quantum computation.

1.1 Problem statement and results

The evolution of a quantum system is determined by its Hamiltonian, which corresponds to a Hermitian matrix H. If we apply a Hamiltonian for time t, then this applies the unitary

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 28; pp. 28:1–28:33

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

gate e^{iHt} to the system. The Hamiltonian of a system is governed by its underlying physics, so oftentimes in quantum computing experiments (e.g. in superconducting qubits) it is easy to apply certain Hamiltonians but not others. From this perspective it is natural to study the computational power of a fixed Hamiltonian H that can be applied to different ordered subsets of qubits for arbitrarily chosen amounts of time. Here we consider the model where we have a fixed two-qubit[1] Hamiltonian H which we can apply to any ordered pair of qubits, where we initialize our system in a computational basis state and perform a computational basis measurement at the end. Now it is natural to ask: What is the computational power of this model for a fixed H? It is known that almost any choice of H in this model yields universal quantum computation [16, 40, 13, 7], but the classification of such universal Hamiltonians remains an open problem. Curiously, there exist subsets of Hamiltonians that do not seem to offer the full power of BQP but nevertheless are hard to simulate classically under plausible complexity assumptions [32, 10, 30].

We focus on a particular family of Hamiltonians H which, even though incapable of universal quantum computation, can perform computations that are hard for classical computers and might offer easier experimental implementation. Specifically, we study Hamiltonians H that can only give rise to mutually commuting gates, so the order in which the gates are applied is irrelevant:

▶ **Definition 1.1.** We say that a two-qubit Hamiltonian H is *commuting* if $[H \otimes I, I \otimes H] = 0$ and $[H \otimes I, I \otimes (THT)] = 0$ and $[H, THT] = 0$, where T is the gate which exchanges two qubits, and $[A, B]$ denotes the quantity $AB - BA$. In other words, H commutes with itself when applied to any pair of qubits.

We are interested in classifying which commuting two-qubit Hamiltonians H allow us to perform computational tasks that are hard for classical computers. In particular, we want to understand when H gives rise to probability distributions which are hard to simulate classically:

▶ **Definition 1.2.** We say that a family of probability distributions $\{\mathcal{D}_x\}_{x\in\{0,1\}^*}$ are *hard to sample from classically* if there exists a constant $c > 1$ such that no BPP machine M can satisfy

$$\frac{1}{c}\Pr[M(x) \text{ outputs } y] \leq \mathcal{D}_x(y) \leq c\Pr[M(x) \text{ outputs } y]$$

for all y in the sample space of $\mathcal{D}_x$.

Clearly, if a commuting H is not capable of creating entanglement from any computational basis state then the system will remain in a product state, so this model will be efficiently classically simulable. Surprisingly, we show that in all the remaining cases H can perform sampling tasks which cannot be simulated classically unless PH collapses.

▶ **Theorem 1.3** (Main Result)**.** *If a commuting two-qubit Hamiltonian H is capable of creating entanglement from a computational basis state, then it gives rise to probability distributions that are hard to sample from classically unless* PH *collapses.*

Additionally, given such an H, our result provides an algorithm which describes the experimental setup required to sample from these hard distributions.

[1] One-qubit Hamiltonians cannot create entanglement, so are efficiently classically simulable in this model.

Experimental implications

Universal quantum computers have proved challenging to implement in practice as they require large overheads for fault-tolerance. As a result, some skeptics have questioned if quantum devices will ever be able to demonstrate an advantage over classical computers [24, 26].

One response to this challenge is to study weaker models of quantum computation which are incapable of universal quantum computation, but still demonstrate an advantage over classical computation [2, 10, 23, 25, 28, 37]. Aliferis et al. [5] have shown that commuting gate sets may be easier to implement fault-tolerantly with superconducting qubits than universal gate sets, and provided numerical evidence that they may admit lower fault-tolerance thresholds. Therefore, commuting computations form a good candidate for providing the first "proof of concept" demonstration of quantum supremacy over classical computation [2]. Our Theorem 1.3 says that almost any commuting Hamiltonian could be used for this demonstration, and additionally provides the experimentalist with a straightforward criterion to determine whether a commuting Hamiltonian can be used to sample from hard distributions.

1.2 Proof ideas

Our proof proceeds in several steps. First, we use that fact that any commuting two-qubit Hamiltonian H is locally diagonalizable:

▶ **Lemma 1.4** ([14, Lemma 33]). *For any commuting two-qubit Hamiltonian there exists a one-qubit unitary U and a diagonal matrix D such that $H = (U \otimes U)D(U^\dagger \otimes U^\dagger)$.*

The proof of this follows from expanding H in the Pauli basis, and deducing relationships between the Pauli coefficients.

Next, we use postselection gadgets to construct a family of one-qubit operations $L(t) : \mathbb{C}^2 \to \mathbb{C}^2$ for $t \in \mathbb{R}$ that that can be applied to the input state using postselection. We then show that these gadgets are universal on a qubit whenever H generates entanglement, so long as H is not some exceptional subcase. The exceptional subcase is $H = X(\theta) \otimes X(\theta)$ where $X(\theta) = \begin{pmatrix} 0 & e^{i\theta/2} \\ e^{-i\theta/2} & 0 \end{pmatrix}$.

▶ **Lemma 1.5.** *If H is capable of creating entanglement from a computational basis state and H is not $X(\theta) \otimes X(\theta)$ for some θ, then it is possible to construct any one-qubit gate by taking products of the $L(t)$ gadgets.*

The main difficulty in proving this fact is that the maps $L(t)$ are in general *non-unitary*. Furthermore since they are generated with postselection, it is unclear how to invert them, so *a priori* they might not even form a group. Fortunately, we find new (and somewhat complicated) postselection gadgets to construct the L^{-1} operations, thus allowing us to apply group-theoretic and Lie-theoretic techniques to address this problem.

The rest of the proof follows from standard techniques in complexity. Since one-qubit gates plus any entangling Hamiltonian form a universal gate set [17, 9], our model can perform universal quantum computation under postselection.

▶ **Lemma 1.6.** *If H is capable of creating entanglement from a computational basis state and H is not $X(\theta) \otimes X(\theta)$ for some θ, then postselected circuits involving H are universal for* BQP.

The proof of this statement uses a non-unitary version of the Solovay-Kitaev theorem proven by Aharonov et al. [4] to show our choice of gate set is irrelevant.

Next, a result of Aaronson [1] tells us that postselecting our circuits further enables us to solve PP-hard problems. It then follows by the complexity arguments put forth by Bremner, Jozsa, and Shepherd [10] and Aaronson [2] that a randomized classical algorithm cannot sample from the probability distributions produced by our circuits unless the polynomial hierarchy collapses.

This completes the classification for all cases except the case $H = X(\theta) \otimes X(\theta)$. Hardness of sampling from these Hamiltonians was previously shown by Fefferman, Foss-Feig, and Gorshkov [18] using a construction which embeds permanents directly in the output distributions of such Hamiltonians. Hardness then follows from the arguments of Aaronson and Arkhipov [2]. We provide a summary of their hardness result for completeness.

1.3 Relation to prior work

Our work is inspired by Bremner, Jozsa, and Shepherd [32, 10, 30], who showed that certain computations with commuting gates are hard to simulate classically unless the polynomial hierarchy collapses. In particular, they show hardness of simulating the gate set comprised of HZH, $(H \otimes H)$(controlled-Z)$(H \otimes H)$, and HPH, where P is the $\pi/8$-phase gate. Similarly, Shepherd [31, 30] considers the power of applying quantum Hamiltonians which are diagonal in the X basis, where the Hamiltonians can be applied only for discrete amounts of time θ; he describes the values of θ for which the resulting circuits are efficiently classically simulable or hard to weakly simulate (that is, to sample from the output probability distribution with a classical computer). Our work differs from these in several ways. First, We consider Hamiltonians rather than gates, and show hardness of *generic* or *average-case* commuting Hamiltonians, rather than showing hardness for worst-case commuting operations. Furthermore, we fully classify the computational complexity of all commuting Hamiltonians, and prove a dichotomy between hardness and classical simulability.

The hardness results we obtain in this paper (as well as those in [10, 31, 30]) are based on the difficulty of sampling the output probability distribution on all n output qubits. A number of other works have considered the power of computations with commuting Hamiltonians, where one only considers the output distribution on a small number of output qubits. For example, Bremner, Jozsa and Shepherd [10] showed that computing the marginal probability distributions on $O(\log(n))$ qubits of their model is in P. Ni and Van den Nest [29] showed that this holds for arbitrary 2-local commuting Hamiltonians, but also showed there exist 3-local commuting Hamiltonians for which this task is hard. Hence the problem of strongly simulating the output distributions (that is, being able to compute the probability of any event) of arbitrary k-local Hamiltonians is hard for $k \geq 3$. Along a similar line of thought, Takahashi et al. [35] showed that there is a system of 5-local commuting Hamiltonians for which weakly simulating the output on $O(\log(n))$ bits is hard.

Additionally, a number of other authors have also considered "weak" models of quantum computation which can sample from difficult probability distributions. Some examples include the one clean qubit model [25, 28], the boson sampling model [2], the quantum fourier sampling model [19], constant depth quantum circuits [37], and temporally unstructured quantum computing [10]. Like many of these models (e.g. [28, 10]), we prove it is difficult for a classical computer to sample from the distribution output by the quantum device with multiplicative error on every output probability. For some of these models, the authors prove stronger hardness results for sampling the output distribution with additive error (as measured in trace distance) [2, 11, 19], but at the cost of making additional complexity-

theoretic assumptions which are not as widely accepted. In comparison with boson sampling, one clean qubit sampling, and quantum fourier sampling, our model has the advantage of possibly having lower fault-tolerance thresholds for implementation [5].

Finally, other works have addressed the classification of universal two-qubit gates and Hamiltonians. Childs, Leung, Mančinska, and Ozols [13] classified the set of two-qubit Hamiltonians which give rise to $SU(4)$ when acting on two qubits, and are hence universal. Lloyd [27] and others [16, 40, 13, 7] have shown that a Haar-random two-qubit gate is universal with probability 1. Our work differs from these in that our Hamiltonians only become universal under postselection. Additionally, Cubitt and Montanaro [14] previously classified the complexity of two-qubit Hamiltonians in the Local Hamiltonian Problem setting. Specifically, given a two qubit Hamiltonian H, they classify the computational complexity of determining the ground state energy of Hamiltonians of the form $\sum_{ij} c_{ij} H_{ij}$ for real coefficients c_{ij}. This is incomparable with our classification, since we are studying the power of the Hamiltonian dynamics (in which the system is not in the ground state), rather than the complexity of their ground states.

2 Preliminaries and statement of Main Theorem

A two-qubit Hamiltonian H is a 4×4 Hermitian matrix. Let T denote the SWAP gate which exchanges two qubits, i.e.

$$T = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

so T maps the state $a|00\rangle + b|01\rangle + c|10\rangle + d|11\rangle$ to the state $a|00\rangle + c|01\rangle + b|10\rangle + d|11\rangle$. Given H, we assume that one can apply either H or THT to any pair of qubits. In other words, we can apply the Hamiltonian oriented from qubit i to qubit j, or from qubit j to qubit i. We will use H_{ij} to denote the Hamiltonian applied from qubit i to qubit j. Additionally, we will assume we can apply $-H$ as well, i.e., we can perform the inverse Hamiltonian.[2]

Suppose we are given some input string $x \in \{0,1\}^n$, and we want to define a distribution on $n' = \mathsf{poly}(n)$ bits which we can efficiently sample from using H. Suppose we initialize a system of n' qubits in a computational basis state $|y\rangle$ for $y \in \{0,1\}^{n'}$, apply each Hamiltonian H_{ij} for time $t_{ij} \in \mathbb{R}$, and then measure all the qubits in the computational basis. (Here the times t_{ij} and the string y may depend on x.) This will induce some probability distribution $\mathcal{D}_x$ over bit strings of length n' on the output bits. Intuitively, these are the sorts of distributions one can efficiently sample from using H, using circuits which start and end in the computational basis.

However, this definition does not quite suffice to capture a realistic model of computation, because we have not specified how the initial state y and the times t_{ij} are chosen. To fix this, we will require that one could use a classical computer to efficiently calculate the experimental setup for each n. In other words, we will require that there exists a polynomial-time algorithm which, given $x \in \{0,1\}^*$, computes the values of y and t_{ij} used

[2] If we had only assumed access to H and positive time evolution, we could always approximate the action of $-H$; this follows from compactness of the unitary group and was shown e.g. in Appendix A of [13]. However, here we are assuming we have exact access to $-H$; this will be useful when arguing about post-selected versions of these circuits.

in the experiment. Furthermore, we will require that the times t_{ij} can be represented with polynomially many bits, and that they are all bounded in magnitude by a polynomial in n. This ensures that as the size of the system grows, the amount of time one needs to run the Hamiltonian does not grow too quickly. In complexity theory this is called a *uniformity* condition. This requirement ensures that any advantage over classical computation arising from this model comes from the power of the quantum computation performed, not the computation of the experimental setup.

This is stated more formally as follows:

▶ **Definition 2.1.** Let $\mathsf{samp\text{-}IQP}(H)$ denote those families of probability distributions $\{\mathcal{D}_x\}$ for which there exists a classical poly-time algorithm $\mathcal{A}$ which, given an input $x \in \{0,1\}^n$, outputs the specifications for a quantum circuit using H whose output distribution family is $\{\mathcal{D}_x\}$. In particular, $\mathcal{A}$ specifies a number of qubits $n' = \mathsf{poly}(n)$, a string $y \in \{0,1\}^{n'}$ and and a series of times $t_{ij} \in \mathbb{R}$, such that running a quantum circuit starting in the state $|y\rangle$, applying the operator $e^{it_{ij}H_{ij}}$ for each (i,j), and then measuring in the computational basis will yield a sample from $\mathcal{D}_x$. Each t_{ij} must be specifiable with $\mathsf{poly}(n)$ bits and be bounded in magnitude by a polynomial in n.

In short, the class $\mathsf{samp\text{-}IQP}(H)$ captures the set of probability distributions one can efficiently sample from using H. In our work, we will show that a classical randomized algorithm cannot sample from this same set of distributions. More precisely, we say that a classical randomized algorithm "weakly simulates" a quantum circuit if its output distribution is close to the output distribution of the quantum circuit. To derive our hardness result, we will consider classical circuits which produce every output with approximately the correct probability, up to multiplicative error:

▶ **Definition 2.2.** A BPP (bounded-error probabilistic polynomial time) machine M *weakly simulates* a family of probability distributions $\{P_x : x \in \{0,1\}^*\}$, where P_x is a distribution over $\{0,1\}^{|x|}$, with multiplicative error $c \geq 1$ if, for all $y \in \{0,1\}^n$,

$$\frac{1}{c}\Pr[M(x) \text{ outputs } y] \leq P(x) \leq c\Pr[M \text{ outputs } y].$$

We can now more precisely state our Main theorem: that our commuting circuits cannot be weakly simulated unless the polynomial hierarchy PH collapses:

▶ **Theorem 2.3** (Main Theorem)**.** *If H is capable of generating entanglement from the computational basis, then* BPP *machines cannot weakly simulate* $\mathsf{samp\text{-}IQP}(H)$ *with multiplicative error $c < \sqrt{2}$ unless* PH *collapses to the third level.*

In other words, there is a dichotomy: either computations which H are efficiently classically simulable, or else they cannot be efficiently simulated unless the polynomial hierarchy collapses. As the non-collapse of the polynomial hierarchy is a widely accepted conjecture in computational complexity, this is strong evidence that $\mathsf{samp\text{-}IQP}(H)$ circuits are not efficiently classically simulable.

2.1 Complexity Preliminaries

Before proceeding to a proof of the Main Theorem, we will introduce some of the complexity-theoretic preliminaries necessary to understand our proof. We assume the reader is familiar with the standard complexity classes such as P, BPP, and NP, as well as oracle notation; for background we refer the reader to Arora and Barak [6] for details. Those readers already

familiar with the complexity theoretic techniques of Bremner, Jozsa, and Shepherd [10] and Aaronson and Arkhipov [2] may wish to skip to the proof of the Main Theorem.

In order to reason about the computational complexity of $\mathsf{samp\text{-}IQP}(H)$ distributions, we will need to introduce the idea of postselected circuits, which we will relate to classical complexity classes such as PP. A postselected quantum circuit is a circuit where one specifies the value of some measurement results ahead of time, and discards all runs of the experiment which do not obtain those measurement outcomes. This is not something one can realistically do in a laboratory setting, because the measurement outcomes you specify may occur extremely infrequently—in fact, they may be exponentially unlikely. However, postselection can help you examine the *conditional probabilities* found in the output distribution of your circuit. In particular, if you can show that those conditional probabilities can encode the answers to very difficult computational problems, then this can provide evidence against the ability to classically simulate such circuits. Therefore, we will now define what it means for a set of probability distributions to decide a problem under postselection. The basic idea is that if some of the conditional probabilities of the system encode the answer to a problem, then we say that problem can be decided by postselected versions of these probability distributions. We define this more formally below:

▶ **Definition 2.4.** Let $\mathsf{PostIQP}(H)$ be the set of languages $L \subseteq \{0,1\}^*$ for which there exists a family of $\mathsf{samp\text{-}IQP}(H)$ circuits $\{\mathcal{D}_x\}$ and a classical poly-time algorithm which, given an input length n, outputs a subset B of qubits and a string $z \in \{0,1\}^{|B|}$ such that

- If $x \in L$, then $\Pr[\ \mathcal{D}_x$ outputs 1 on its first bit $\mid$ bits B take value $z\ ] \geq 2/3$.
- If $x \notin L$, then $\Pr[\ \mathcal{D}_x$ outputs 1 on its first bit $\mid$ bits B take value $z\ ] \leq 1/3$.

In other words, there exists a poly-time algorithm which outputs an experimental setup and a postselection scheme such that the conditional probabilities of $\mathcal{D}_x$ encode the answer to the problem. In general, the choice of constants $1/3$ and $2/3$ in the above definition might matter. For instance, when $\mathsf{PostIQP}(H)$ is not capable of universal classical computation, it is unclear how to take the majority vote of many repetitions to amplify the success probability. However, we only consider the class $\mathsf{PostIQP}(H)$ in cases where $\mathsf{PostIQP}(H)$ can perform universal classical computation, and thus the choice of constants $1/3$ and $2/3$ is arbitrary.

One can likewise define the classes $\mathsf{PostBQP}$ and $\mathsf{PostBPP}$[3] which capture the power of postselected quantum computation and postselected randomized computation, respectively.

Finally, we introduce the polynomial hierarchy. The ith level of the polynomial hierarchy, denoted Δ_i, is defined as follows: let $\Delta_1 = \mathsf{P}$, let $\Delta_2 = \mathsf{P}^{\mathsf{NP}}$, let $\Delta_3 = \mathsf{P}^{\mathsf{NP}^{\mathsf{NP}}}$, let $\Delta_4 = \mathsf{P}^{\mathsf{NP}^{\mathsf{NP}^{\mathsf{NP}}}}$, and so on. Here, we write A^B to refer to computations that can be performed with an A machine which has been augmented with the ability to solve problems in B in a single timestep. The polynomial hierarchy, denoted PH, is defined as $\mathsf{PH} = \bigcup_{i \in \mathbb{N}} \Delta_i$. It is widely conjectured that each level of the polynomial hierarchy is distinct; in other words, $\Delta_i \subsetneq \Delta_{i+1}$ for all $i \in \mathbb{N}$. This can be seen as a generalization of the conjecture that $\mathsf{P} \neq \mathsf{NP}$.

One of the main technical tools we will use in our proof is the following lemma, which was first shown by Bremner, Jozsa, and Shepherd [10], but which we will make extensive use of in our paper:

▶ **Lemma 2.5.** *Suppose that* $\mathsf{PostBQP} \subseteq \mathsf{PostIQP}(H)$ *for some* H. *Then* BPP *machines cannot weakly simulate* $\mathsf{samp\text{-}IQP}(H)$ *with multiplicative error* $c < \sqrt{2}$ *unless* PH *collapses to the third level.*

[3] $\mathsf{PostBPP}$ is more commonly known as $\mathsf{BPP}_{\mathsf{path}}$.

In other words, if postselected commuting Hamiltonian circuits are capable of performing (postselected) universal quantum computation, then they cannot be weakly simulated by a classical computer under plausible complexity assumptions. The fundamental reason this is true is that the class PostBQP is substantially more powerful than the class PostBPP. In particular, Aaronson [1] showed that PostBQP = PP. Here PP (which stands for Probabilistic Polynomial-time) is the set of languages L decidable by a poly-time randomized algorithm M, such that

- If $x \in L$, then $\Pr[M(x) \text{ accepts}] > 1/2$;
- otherwise, $\Pr[M(x) \text{ accepts}] \leq 1/2$.

In other words, the class PP represents the class of problems solvable by randomized algorithms, where the probability of acceptance of "yes" and "no" instances is different, but may only differ by an exponentially small amount[4]. A famous result in complexity, known as Toda's Theorem [38], states that $\mathsf{PH} \subseteq \mathsf{P}^{\mathsf{PP}}$. In other words, the class PP is nearly as powerful as the entire polynomial hierarchy.

On the other hand, the class PostBPP is far weaker; it lies in the third level of the polynomial hierarchy. So if one assumes that PH does not collapse to the third level, then PostBPP $\neq$ PostBQP; i.e. PostBQP is a stronger complexity class than PostBPP.

From this, we can now state why the inclusion PostBQP $\subseteq$ PostIQP(H) implies there cannot exist an algorithm to simulate PostIQP(H) circuits. Suppose there were a BPP algorithm to weakly simulate such circuits. Then, by postselecting this BPP algorithm, we could solve a PostBQP-hard problem in PostBPP, which would imply the collapse of the polynomial hierarchy. A more formal statement of this proof is given below:

Proof of Lemma 2.5. The proof of this corollary is given in [10] Theorem 2 and Corollary 1, but we provide a summary for completeness. Suppose that a BPP machine M can weakly simulate samp-IQP(H) circuits to multiplicative error $c < \sqrt{2}$. Then for any individual output string x, we have $\frac{1}{c}\Pr[M \text{ outputs } x] \leq P(x) \leq c\Pr[M \text{ outputs } x]$. Since PostBQP $\subseteq$ PostIQP(H), and PostBQP = PP [1], this can be shown to imply PP $\subseteq$ PostBPP. But PostBPP $\subseteq$ PostBQP = PP, so this implies PostBPP = PP. Hence by Toda's theorem [38], we have $\mathsf{PH} \subseteq \mathsf{P}^{\mathsf{PP}} = \mathsf{P}^{\mathsf{PostBPP}} \subseteq \Delta_3$, where Δ_3 is the third level of the polynomial hierarchy. Hence $\mathsf{PH} = \Delta_3$ as claimed. ◀

Note that in certain cases, Fujii et al. [20] showed that this hardness of simulation result could be improved to imply the collapse of PH to the second level rather than the third, using a different complexity-theoretic argument involving the class NQP. However, their argument is gate-set dependent, so does not apply to our model for arbitrary commuting Hamiltonians.

We now proceed to a proof of the Main Theorem.

3 Proof of Main Theorem

The basic idea is to use postselection gadgets to show that postselected samp-IQP(H) circuits are capable of performing universal quantum computation. Hence, adding further postselections allows one to decide any language in PostBQP. By Lemma 2.5, this proves hardness of weakly simulating such circuits unless PH collapses.

[4] Note the difference is probabilities is always at least $2^{-\text{poly}(n)}$, because a PP algorithm can only make polynomially many coin flips.

Proof of Theorem 2.3. Suppose we have a commuting two-qubit Hamiltonian H. The first step in our proof is to characterize the structure of such H. It is clear that if H is diagonal under a local change of basis, i.e. $H = (U \otimes U)D(U^\dagger \otimes U^\dagger)$ for some one-qubit $U \in SU(2)$ and diagonal matrix D, then H is commuting. However, it is possible *a priori* that there exist commuting Hamiltonians which are not of this form. If T is the gate that swaps two qubits, then the fact that H is commuting implies that $H \otimes I$, $(THT) \otimes I$, $I \otimes H$, and $I \otimes (THT)$ are all simultaneously diagonalizable. However, it might be that this simultaneous diagonalization can only happen under a non-local change of basis. Fortunately, it turns out this is not possible - any commuting Hamiltonian must be locally diagonalizable. This was first shown by Cubitt and Montanaro [14].

▶ **Claim 3.1** ([14, Appendix B, Lemma 33]). *If H is a 2-local commuting Hamiltonian, then $H = (U \otimes U)D(U^\dagger \otimes U^\dagger)$ for some one-qubit $U \in SU(2)$ and diagonal matrix D.*

We provide a proof of Claim 3.1 in Appendix A, which uses expansion in the Pauli basis. One can also prove this fact using linear algebra, but the proof becomes complicated in the case of degenerate eigenvalues. We thank Jacob Taylor for pointing us to this simplified proof, and Ashley Montanaro for pointing us to the proof in reference [14].

By Claim 3.1, we know that $H = (U \otimes U)\,\mathrm{diag}(a, b, c, d)(U^\dagger \otimes U^\dagger)$ for some one-qubit unitary $U = \begin{pmatrix} \alpha & -\beta^* \\ \beta & \alpha^* \end{pmatrix}$ and some real parameters a, b, c, d. The trace of H contributes an irrelevant global phase to the unitary operator it generates, so without loss of generality we can assume H is traceless, i.e., $a + b + c + d = 0$.

Note that if $a = d = -1$, $b = c = 1$, and $|\alpha| = |\beta|$, then we have that $H = X(\theta) \otimes X(\theta)$, where $e^{i\theta} = \alpha/\beta$. As mentioned previously, these Hamiltonians are hard to simulate by an independent hardness result of Fefferman et al. [18], so in the rest of our proof, we will assume we are not in the case $a = d = -1$, $b = c = 1$ and $|\alpha| = |\beta|$. For completeness we will provide a summary of their work at the end of this proof.

We now consider the conditions under which computations with H are efficiently classically simulable. First, if H is diagonal in the computational basis, then it is obviously classically simulable, because it cannot generate entanglement from the computational basis. This corresponds to the case that $\alpha = 0$ or $\beta = 0$. So we can assume for the result of the proof that $\alpha \neq 0$ and $\beta \neq 0$.

Another way that H can fail to generate entanglement from the computational basis is if $b + c = a + d$. Since we are assuming the Hamiltonian is traceless this is equivalent to the condition $b + c = 0$. Indeed if H satisfies $b + c = 0$, and H is traceless so $a + d = 0$, then it is easy to check that e^{iHt} is nonentangling for all $t \in \mathbb{R}$. So we can assume in the rest of the proof that $b + c \neq 0$.

We now show that for all remaining H, we have $\mathsf{PostBQP} \subseteq \mathsf{PostIQP}(H)$. To do so, we break into two cases. Either $b = c$, so $H = THT$ and the Hamiltonian is identical when applied from qubit 1 to 2 vs. from 2 to 1, or $b \neq c$ so $H \neq THT$. For clarity of presentation, we will prove our main theorem in the case $b = c$, as this proof uses simpler notation. An analogous proof holds for the case $b \neq c$, which we provide in Appendix D.

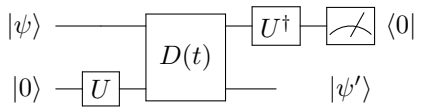
Now in the case $b = c$, consider the rescaled Hamiltonian $H' = H/b$. Since $b + c \neq 0$ and $b = c$ this Hamiltonian is well-defined, and we have $H' = (U \otimes U)\,\mathrm{diag}(a', 1, 1, d')(U^\dagger \otimes U^\dagger)$ for some real parameters a' and d' which obey $a' + d' = -2$. Now consider the two-qubit unitary $V(t)$ we obtain from running H' for time $t \in \mathbb{R}$

$$V(t) = e^{itH'} = (U^{\otimes 2})D(t)({U^\dagger}^{\otimes 2}),$$

where $D(t) \triangleq \mathrm{diag}(e^{ia't}, e^{it}, e^{it}, e^{id't})$. Here we have used the fact that if U is an arbitrary unitary, then $e^{UH'U^\dagger} = Ue^{H'}U^\dagger$.

A samp-IQP(H) circuit is specified by times t_{ij} for all unordered[5] pairs of qubits (i, j), as well as an initial basis state $|y\rangle$ for $y \in \{0,1\}^{\mathrm{poly}(n)}$. The circuit consists of applying $V(t_{ij})$ to each pair of qubits (i, j) to $|y\rangle$, and then measuring in the computational basis. This can be easily seen to be equivalent to the following circuit: Start in the state $|y\rangle$, apply U to every qubit, then apply $D(t_{ij})$ to each pair of qubits; finally, apply $U^\dagger$ to every qubit and measure in the computational basis. (This is true because all factors of U and $U^\dagger$ in the circuit cancel except those at the beginning and end).

We will now show how to make post-selected gates of this form perform universal quantum computing. The basic idea is that we already have a two-qubit entangling Hamiltonian at our disposal. Therefore, if we could show how to perform arbitrary one-qubit gates using post-selection, this would form a universal gate set for quantum computing by the result of Dodd et al. [17] or Bremner et al. [9]. Following the method of Bremner, Jozsa, and Shepherd [10], we consider the following post-selection gadget, denoted $L(t)$, which performs an operation on a single qubit state $|\psi\rangle$:



The postselection is denoted in the circuit by $\langle 0|$. Note that this gadget preserves the property that every line begins with $U|0\rangle$, and ends with $U^\dagger$ and a measurement. Hence, if we could use these postselection gadgets to perform arbitrary single-qubit gates, then we could perform universal quantum computing under postselection as follows: Given a target quantum circuit to simulate, compile the circuit out of gates of the form $D(t)$ and single-qubit gates. Additionally, add a $UU^\dagger$ (which is the identity) at the beginning and end of every line, so that each line starts with a U and ends with a $U^\dagger$. Now this circuit consists of applying a column of U's, then a series of diagonal gates $D(t)$ and one-qubit gates, followed by a column of $U^\dagger$s. This almost has the form of a samp-IQP(H) circuit, with the exception of the one-qubit gates (note that these include both the gates $U^\dagger$ in the second column and the gates U in the second to last column). Now for each one-qubit gate g, replace it with its implementation using postselection gadgets $L(t)$. After this transformation, each line begins with a U, ends with a $U^\dagger$, and contains only diagonal gates $D(t)$ in the interior of the circuit. However, now we've additionally specified some postselection bits, so we have created a PostIQP(H) circuit which simulates universal quantum computing.

Let us examine what transformation $L(t)$ actually performs on the qubits involved. The gadget performs some linear transformation on the input state $|\psi\rangle$. In particular, it acts on $|\psi\rangle$ by

$$L(t) = \frac{1}{|\alpha||\beta|\sqrt{-2i\sin(2t)}} \begin{pmatrix} |\alpha|^2 e^{ia't} & \alpha\beta^* e^{it} \\ \alpha^*\beta e^{it} & |\beta|^2 e^{id't)} \end{pmatrix}.$$

This is a non-unitary transformation, so it does not preserve the norms of vectors. Since we only care about how $L(t)$ behaves on the projective Hilbert space of quantum states, we can choose the overall normalization so that $L(t) \in SL(2, \mathbb{C})$. Note that this operator is well-defined only if the denominator above is non-zero, so we will require that $t \in (0, \pi) \cup (\pi, 2\pi)$.

[5] This is because we are considering the case $b = c$ i.e. $H = THT$.

In addition to being able to perform the transformation $L(t)$ as t ranges over $t \in (0,\pi)\cup(\pi,2\pi)$, we can also perform products of such transformations. In fact, we can perform any operation in the set

$$S \triangleq \overline{\langle \{L(t) : t \in (0,\pi)\cup(\pi,2\pi)\}\rangle}.$$

Here the angled brackets $\langle A\rangle$ denote the set of all matrices obtained by finite products of elements of A. The bar above $\overline{\langle A\rangle}$ means that we take the closure of this set in $SL(2,\mathbb{C})$; in other words, we include all matrices that one can obtain by taking limits of sequences of finite products of A, so long as the limit point belongs to $SL(2,\mathbb{C})$.

If the matrices $L(t)$ were in a compact space such as $SU(2)$, then it would immediately follow that S contains inverses of all its elements.[6] Therefore we would know that S is a group, and we could apply tools from group theory to categorize S. However, our matrices are in the non-compact space $SL(2,\mathbb{C})$. Therefore it is not clear whether S is closed under taking inverses, so S *might not be a group*! Furthermore, since L is obtained under post-selection, the assumption that we can perform the inverse of H does not imply we can perform L^{-1}.

To fix this problem, we find additional gadgets which allow us to construct L^{-1} by adding additional postselections to our circuit. In particular, we will show that for each $L(t)$, there exists a postselection gadget of finite size which performs $L(t)^{-1}$ exactly. An important restriction on this construction is that this inverse must be efficiently computable. Specifically, for each $L(t)$ the size of the postselection gadget required to invert $L(t)$ is of constant size. Additionally, the construction of the postselection gadget will in general contain several time parameters which one needs to set in order to obtain $L(t)^{-1}$. We also require that we can set these times so that we obtain L^{-1} to accuracy ϵ in $\text{polylog}(k_L 1/\epsilon)$ time, where k_L is a constant which depends on $L(t)$ only. Furthermore, the amount of time needed to run the Hamiltonians in the inverse gadget are bounded above by a polynomial. For convenience we will refer to these properties as "the construction is efficiently computable."

At first glance it might sound like this definition of "efficiently computable" is too weak, because the inverses of arbitrary L matrices might require large postselection gadgets. However, later in our construction we will use the fact that for any fixed Hamiltonian H, we will only need to invert a finite set of L matrices. Hence for fixed H, the size of the postselection gadgets which appear in our circuit will be upper bounded by a constant depending on H only, but not on the size of the problem we are solving under postselection. Furthermore, for fixed H, we can compute the times in the inversion gadgets to invert the relevant L matrices to exponential accuracy in polynomial time. This ability to invert the L matrices to exponential precision will later be crucial for our hardness of sampling result.

Furthermore, note that in the case that $H \neq THT$, the construction of these gadgets can be made substantially simpler. In particular, the gadgets to construct $L^{-1}(t)$ are of size 4 for any t, and the times used in running the Hamiltonians are trivially efficiently computable to polynomial digits of accuracy. From a practical experimental perspective these circuits would be easier to construct, and since $H \neq THT$ is the generic case for commuting Hamiltonians, would be applicable for almost all commuting Hamiltonians. We include this construction in Appendix D.

[6] To see this, take an element $s \in S$. If s has finite order, than its inverse is clearly in S. If s has infinite order, consider the sequence $1, s, s^2, \ldots$. Since the matrices are in a compact space T, the sequence of powers must have a convergent subsequence, i.e. there must be positive $n_1, n_2, n_3 \ldots$ such that $n_1 < n_2 < \ldots$ and $s^{n_1}, s^{n_2}, \ldots$ approach some element $t \in T$. Therefore the sequence $s^{n_2-n_1}, s^{n_3-n_2}, \ldots$ must approach the identity, and the sequence $s^{n_2-n_1-1}, s^{n_3-n_2-1}, \ldots$ must approach s^{-1}.

▶ **Claim 3.2.** *For any given $L(t)$, where $t \in (0,\pi) \cup (\pi, 2\pi)$, it is possible to construct $L(t)^{-1}$ by introducing a constant number of postselections and a constant number of ancillas into the circuit. Furthermore, this construction is efficiently computable in the manner described above.*

The proof of Claim 3.2 can be found in Appendix B, and is somewhat involved.

We now redefine S so that its base set contains these inverses:

$$S \triangleq \overline{\langle \{L(t) : t \in (0,\pi) \cup (\pi,2\pi)\} \cup \{L(t)^{-1} : t \in (0,\pi) \cup (\pi,2\pi)\}\rangle}.$$

Using this definition, we can now show using standard techniques that S is a Lie group—this is essentially a consequence of Cartan's closed subgroup theorem [12] and the fact that inversion is a continuous operation in the matrix entries on $SL(2,\mathbb{C})$. Once we know that S has the structure of a Lie group, we can apply the theory of Lie algebras to identify what set of matrices are in S. In particular, we can show that S generates all of $SL(2,\mathbb{C})$.

▶ **Claim 3.3.** $S = SL(2,\mathbb{C})$.

The proof of this claim is a tedious but straightforward calculation using Lie algebras and properties of the exponential map on $SL(2,\mathbb{C})$. The proof uses the fact that we are not in one of the cases excluded by our theorem (i.e. H does generate entanglement and is not $X(\theta) \otimes X(\theta)$ for some θ) - in these cases one does *not* find that $S = SL(2,\mathbb{C})$ as one would expect. In certain special cases, the gadgets $L(t)$ alone do not generate $SL(2,\mathbb{C})$, specifically when $a' = \pm 1$ or $a' = -3$. In these cases, we show that one can add additional postselection gadgets, which are closed under taking inverses, which boost the power of the $L(t)$ transformations to cover all of $SL(2,\mathbb{C})$. This simply reflects that for very particular Hamiltonians, our L matrices need additional help to span all 1-qubit operations. We include the proof in Appendix C.

Now that we have shown density in $SL(2,\mathbb{C})$, as well as the fact that we can produce inverses of the gates in our generating set, our proof of yielding PP under postselection follows almost immediately. In particular, we will invoke the following theorem by Aharanov, Arad, Eban and Landau [4]:

▶ **Theorem 3.4** ([4, Theorem 7.6], adapted to our case)**.** *There exists a constant $\epsilon_0 > 0$ such that, for any $G = \{g_1 \dots g_k\} \subset SL(2,\mathbb{C})$ which is an ϵ_0-net over B, where B is the set of operations in $SL(2,\mathbb{C})$ which are 2.1-far from the identity (which in particular contains $SU(2)$), then for any unitary $U \in SU(2,\mathbb{C})$, there is an algorithm to find an ϵ-approximation to U using $\mathsf{polylog}(1/\epsilon)$ elements of G and their inverses which runs in $\mathsf{polylog}(1/\epsilon)$ time.*

In the above theorem, when we say an operation is "ϵ-far" from another, we are referring to the operator norm.

From this, we can immediately prove the main theorem. Suppose we wish to compute a language $L_0 \in \mathsf{PP}$, and we have a commuting Hamiltonian H of the form promised in Theorem 2.3. By Aaronson's result that $\mathsf{PP} \subseteq \mathsf{PostBQP}$ [1], there is an efficiently computable postselected quantum circuit C composed of Hadamard and Toffoli gates which computes L. Additionally, by Claim 3.3 there exists a finite set G of products of L's and L^{-1}'s which form an ϵ_0-net over B (which can be computed in finite time). Hence by Theorem 3.4 there is a poly-time algorithm which expresses single-qubit gates as products of elements of G to exponential accuracy. Likewise, since H is entangling, we can generate some entangling two-qubit gate g, as well as its inverse g^{-1} (by applying $-H$). Since g and single-qubit gates are universal [9], by the usual Solovay–Kitaev theorem [15], we can express the circuit C in terms of g, g^{-1}, and single-qubit gates to exponential accuracy with polynomial overhead.

Combining these, we can express the circuit C as a polynomial sized product of g's, g^{-1}'s, L's, and L^{-1}'s , which we can express as a $\mathsf{PostIQP}(H)$ circuit using the gadgets described previously. Hence this $\mathsf{PostIQP}(H)$ circuit decides the language L_0.

Note that in this construction, it is crucial that we only ever need to invert a finite number of $L(t)$ matrices. This ensures that the size of the postselection gadgets involved to construct the L^{-1} operations are upper bounded by a constant depending on the choice of H only. Additionally, it is important that we can construct the L^{-1} matrices exponential accuracy. This is crucial because in order to perform $\mathsf{PostBQP}$ under postselection, one needs to be able to simulate Aaronson's algorithm to exponential accuracy [7]. Fortunately our construction allows us to simulate the algorithm to high accuracy, and hence these Hamiltonians can be used to sample from probability distributions which are not possible to simulate with a classical computer unless the polynomial hierarchy collapses.

This completes the proof in all cases except the exceptional case $H = X(\theta) \otimes X(\theta)$. This has a separate hardness of sampling result which was shown by Fefferman, Foss-Feig, and Gorshkov [18]. In particular, they showed the following:

▶ **Theorem 3.5** (Fefferman et al. [18]). *If $H = X(\theta) \otimes X(\theta)$ for some θ, then a* BPP *machine cannot weakly simulate* $\mathsf{samp\text{-}IQP}(H)$ *with any constant multiplicative error unless* PH *collapses to the third level.*

Their proof makes use of that fact that using such Hamiltonians, for any matrix $A \in \{0, \pm 1\}^n$, one can perform a unitary U on a system of $O(n)$ qubits such that $\langle 1^n|U|0^n\rangle = k\,(\mathrm{Perm}(A) + \epsilon)$, where k is independent of A and exponentially small in n, $\mathrm{Perm}(A)$ denotes the permanent of A, and ϵ is a term with norm $o(2^{-n})$. Note that $\mathrm{Perm}(A)^2$ is $\#\mathsf{P}$-hard to compute with any constant multiplicative error [2]. Therefore Theorem 3.5 immediately follows by the techniques of Aaronson and Arkhipov [2] - because if there were an efficient classical simulation of such circuits, then using approximate counting [34], one could approximate $\mathrm{Perm}(A)^2$ to multiplicative error $\left(1 + \frac{1}{\mathsf{poly}(n)}\right)$ in $\mathsf{BPP}^{\mathsf{NP}}$. But $\mathsf{BPP}^{\mathsf{NP}} \subseteq \Delta_3$, so again by Toda's theorem [38] this implies the collapse of PH to the third level.

This completes the last remaining case, and hence completes the proof. ◀

4 Open Problems

Our results leave a number of open problems.

1. An interesting open problem is to classify *all* Hamiltonians in terms of their computational power under this model. Childs et al. [13] previously classified which two-qubit Hamiltonians can perform any unitary on two qubits. However, this does not classify which Hamiltonians are computationally universal for two reasons. First, as Childs et al. point out in their paper, it is possible that H fails to generate all unitaries on two qubits, but does generate all unitaries on three qubits (i.e. adding ancillas helps one attain universality). It remains open to classify which two-qubit H generate all unitaries on sufficiently large systems. Second, even if a Hamiltonian H does not generate all unitaries, it is still possible that H is computationally universal. For example, H could be universal on an encoded subspace. Classifying which Hamiltonians are universal under an encoding seems to be a challenging task. We conjecture that the power of any two-qubit Hamiltonian obeys a dichotomy: either H is efficiently classicaly simulable in this model,

[7] This is because the algorithm postselects on an exponentially unlikely event, so to maintain polynomial accuracy after postselection we require exponential accuracy prior to postselection.

or it is universal under postselection and hence cannot be weakly simulated unless PH collapses. This is true of all known two-qubit Hamiltonians, and our classification proves this result rigorously in the case of commuting Hamiltonians.

2. In this paper we considered the power of quantum circuits with commuting Hamiltonians. A more difficult related problem is classify the power of quantum circuits with commuting gate sets. The challenge in solving this problem would be to classify when a discrete set of L's generates a continuum of gates. There are some sufficient conditions under which this holds (see e.g. Aharonov et al. [4], Corollary 9.1). However, finding necessary and sufficient conditions under which a finite set of operators densely generates a continuous subgroup of $SL(2,\mathbb{C})$ seems very difficult, in part because there is no complete, explicit classification of discrete subgroups of $SL(2,\mathbb{C})$. Indeed, discrete subgroups of $SL(2,\mathbb{C})$ are related to the theory of Möbius transformations [8], where they are known as "Kleinian subgroups," and they are the subject of a deep area of mathematical research.

Acknowledgements. We thank Bill Fefferman, Michael Foss-Feig, and Alexey Gorshkov for allowing us to include a summary of their unpublished work [18]. We also thank Jacob Taylor for pointing us to a simplified proof of Claim 3.1, Michael Bremner for pointing us to references [30, 31], and Scott Aaronson, Joseph Fitzsimons and Ashley Montanaro for helpful discussions. A.B. was supported in part by the National Science Foundation Graduate Research Fellowship under Grant No. 1122374, in part by the Center for Science of Information (CSoI), an NSF Science and Technology Center, under grant agreement CCF-0939370, and in part by Scott Aaronson's NSF Waterman award. L.M. was supported by Singapore Ministry of Education (MOE) and National Research Foundation Singapore, as well as MOE Tier 3 Grant "Random numbers from quantum processes" (MOE2012-T3-1-009). X.Z. was supported by the MIT UROP program.

References

1 Scott Aaronson. Quantum computing, postselection, and probabilistic polynomial-time. *Proc. R. Soc. A*, page 0412187, 2005.

2 Scott Aaronson and Alex Arkhipov. The Computational Complexity of Linear Optics. *Theory of Computing*, 9(4):143–252, 2013.

3 Dorit Aharonov and Itai Arad. The BQP-hardness of approximating the Jones Polynomial. *New J. Phys.* 13 035019, 2011.

4 Dorit Aharonov, Itai Arad, Elad Eban, and Zeph Landau. Polynomial Quantum Algorithms for Additive approximations of the Potts model and other Points of the Tutte Plane. arxiv:quant-ph/0702008, 2007.

5 Panos Aliferis, Frederico Brito, David P. DiVincenzo, John Preskill, Matthias Steffen, and Barbara M. Terhal. Fault-Tolerant Computing With Biased-Noise Superconducting Qubits. *New J. Phys.* 11 013061, 2009.

6 Sanjeev Arora and Boaz Barak. Computational Complexity: A Modern Approach. *Cambridge University Press*, New York, NY, USA, 1st edition, 2009.

7 Bela Bauer, Claire Levaillant, and Michael Freedman. Universality of single quantum gates. arXiv:1404.7822v3

8 Alan F. Beardon. The geometry of discrete groups. *Springer-Verlag*, Berlin, 1983.

9 Michael J. Bremner, Christopher M. Dawson, Jennifer L. Dodd, Alexei Gilchrist, Aram W. Harrow, Duncan Mortimer, Michael A. Nielsen, and Tobias J. Osborne. A practical scheme for quantum computation with any two-qubit entangling gate. *Phys. Rev. Lett.* 89, 247902, 2002.

10 Michael J. Bremner, Richard Jozsa, and Dan J. Shepherd. Classical simulation of commuting quantum computations implies collapse of the polynomial hierarchy. *Proc. R. Soc. A* 467, 459, 2011.

11 Michael J. Bremner, Ashley Montanaro, and Dan J. Shepherd. Average-case complexity versus approximate simulation of commuting quantum computations. arXiv:1504.07999, 2015

12 Élie Cartan. La théorie des groupes finis et continus et l'analysis situs. Mémorial des Sciences Mathématiques, 42:1–61, 1930.

13 Andrew M. Childs, Debbie Leung, Laura Mančinska, and Māris Ozols. Characterization of universal two-qubit Hamiltonians. *Quantum Info. Comput.* 11, 19–39, 2011.

14 Toby Cubitt and Ashley Montanaro. Complexity classification of local Hamiltonian problems. *Proc. IEEE FOCS* 2014.

15 Christopher M. Dawson and Michael A. Nielsen. The Solovay-Kitaev algorithm. *Quantum Info. Comput.*, 6(1):81–95, January 2006.

16 David Deutsch, Adriano Barenco, and Artur Ekert. Universality in quantum computation. *Proc. R. Soc. A*:449, 669-677, 1995.

17 Jennifer L. Dodd, Michael A. Nielsen, Michael J. Bremner, and Robert T. Thew. Universal quantum computation and simulation using any entangling Hamiltonian and local unitaries. *Phys. Rev. A (Rapid Comm.)* 65, 040301(R), 2002.

18 Bill Fefferman, Michael Foss-Feig and Alexey Gorshkov. Unpublished Manuscript.

19 Bill Fefferman and Chris Umans. The Power of Quantum Fourier Sampling. arXiv:1507.05592, 2015.

20 Keisuke Fujii, Hirotada Kobayashi, Tomoyuki Morimae, Harumichi Nishimura, Shuhei Tamate, and Seiichiro Tani. Impossibility of Classically Simulating One-Clean-Qubit Computation. arXiv:1409.6777, 2014.

21 Brian Hall. Lie Groups, Lie Algebras, and Representations: An Elementary Introduction. *Springer-Verlag*, New York, 2003.

22 Yenjo Han, Lane A. Hemaspaandra, and Thomas Thierauf. Threshold Computation and Cryptographic Security. *SIAM J. Comput.*, 26(1), 59–78, 1997.

23 Stephen P. Jordan. Permutational quantum computing. *Quantum Info. Comput.*, 10(5):470–497, 2010.

24 Gil Kalai. How Quantum Computers Fail: Quantum Codes, Correlations in Physical Systems, and Noise Accumulation. arXiv:1106.0485, 2011.

25 Emanuel Knill and Raymond Laflamme. Power of One Bit of Quantum Information. *Phys. Rev. Lett.* 81(25):5672–5675, 1998.

26 Leonid A. Levin. Polynomial time and extravagant models, in The tale of one-way functions. Problems of Information Transmission, 39(1):92-103, 2003. arXiv:cs.CR/0012023.

27 Seth Lloyd. Almost any quantum logic gate is universal. *Phys. Rev. Lett.* 10, pp. 346–349, 1995.

28 Tomoyuki Morimae, Keisuke Fujii, and Joseph F. Fitzsimons. On the hardness of classically simulating the one clean qubit model. *Phys. Rev. Lett.* 112, 130502, 2014.

29 Xiaotong Ni and Maarten Van den Nest. Commuting quantum circuits: efficient classical simulations versus hardness results. *Quantum Info. Comput.* 13:1–2, 0054–0072, 2013.

30 Dan Shepherd. Quantum Complexity: restrictions on algorithms and architectures. PhD Thesis, University of Bristol, 2009.

31 Dan Shepherd. Binary Matroids and Quantum Probability Distributions. arXiv:1005.1744 (2010).

32 Dan Shepherd and Michael J. Bremner. Temporally unstructured quantum computation. *Proc. R. Soc. A*:465, 1413, 2009.

33 Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring *Proc. IEEE FOCS'94*, pp. 124–134, 1994.

34 Larry Stockmeyer. The complexity of approximate counting. *Proc. STOC'83*, pp. 118–126, 1983.

35 Yasuhiro Takahashi, Seiichiro Tani, Takeshi Yamazaki, and Kazuyuki Tanaka. Commuting Quantum Circuits with Few Outputs are Unlikely to be Classically Simulatable. Quantum Inf. Comp. 16:3&4, pp. 251–270, 2016.

36 Barbara M. Terhal and David P. DiVincenzo. Classical simulation of noninteracting-fermion quantum circuits. Phys. Rev. A. 65.032325, 2002.

37 Barbara M. Terhal and David P. DiVincenzo. Adaptive Quantum Computation, Constant Depth Quantum Circuits and Arthur-Merlin Games. Quant. Inf. Comp. 4:2, pp. 134–145, 2004.

38 Seinosuke Toda. PP is as hard as the polynomial-time hierarchy. *SIAM J. Comp.* 20(5), pp. 865–877, 1991.

39 Leslie Valiant. Quantum circuits that can be simulated classically in polynomial time. SIAM Journal on Computing, 31(4):1229–1254, 2002.

40 Nik Weaver. On the universality of almost every quantum logic gate. *J. Math. Phys.* 41, 240, 2000.

A Commuting Hamiltonians are locally diagonalizable

To establish Claim 3.1, we prove the following stronger statement.

▶ **Claim A.1.** *If H is a two-qubit Hamiltonian and $[H\otimes I, I\otimes H]=0$, then $(U\otimes U)H(U\otimes U)^\dagger$ is diagonal for some one-qubit unitary U.*

This is actually slightly stronger than Lemma 33 of [14], which shows that if $[H\otimes I, I\otimes H] = [H\otimes I, I\otimes THT] = [THT\otimes I, I\otimes H] = 0$, then H is locally diagonalizable. Here we merely require that $[H\otimes I, I\otimes H]=0$.

Proof. As a first step we expand H in Pauli basis and let α_{AB} be the coefficient at $A\otimes B$ term for any $A,B\in\{I,X,Y,Z\}$. Also, for all $A\in\{I,X,Y,Z\}$, let

$$\vec{c}_A := (\alpha_{XA},\alpha_{YA},\alpha_{ZA})^T \quad \text{and} \quad \vec{r}_A := (\alpha_{AX},\alpha_{AY},\alpha_{AZ})^T. \tag{1}$$

Given a vector $\vec{v}=(v_x,v_y,v_z)^T\in\mathbb{R}^3$, we adopt a commonly used notation and write $\vec{v}\cdot\vec{\sigma}$ to denote the linear combination $v_xX+v_yY+v_zZ$.

Since $(H\otimes I)(I\otimes H) = (I\otimes H)(H\otimes I)$, we know that both products must have the same expansion in Pauli basis. Let us fix $A,B\in\{I,X,Y,Z\}$ and consider the terms of the form $A\otimes\_\otimes B$ in the Pauli expansion of each of the products.

First, for $(H\otimes I)(I\otimes H)$ we notice that, when restricted to terms of the form $A\otimes\_\otimes B$, its Pauli expansion is given by

$$\big(A\otimes(\alpha_{AI}I+\vec{r}_A\cdot\vec{\sigma})\otimes I\big)\big(I\otimes(\alpha_{IB}I+\vec{c}_B\cdot\vec{\sigma})\otimes B\big) = \tag{2}$$

$$A\otimes\big(\alpha_{AI}\alpha_{IB}I+(\alpha_{AI}\vec{c}_B+\alpha_{IB}\vec{r}_A)\cdot\vec{\sigma}+(\vec{r}_A\cdot\vec{\sigma})(\vec{c}_B\cdot\vec{\sigma})\big)\otimes B = \tag{3}$$

$$A\otimes\big((\alpha_{AI}\alpha_{IB}+\vec{r}_A\cdot\vec{c}_B)I+(\alpha_{AI}\vec{c}_B+\alpha_{IB}\vec{r}_A+i(\vec{r}_A\times\vec{c}_B))\cdot\vec{\sigma}\big)\otimes B, \tag{4}$$

where we have applied the identity $(\vec{v}\cdot\vec{\sigma})(\vec{w}\cdot\vec{\sigma}) = (\vec{v}\cdot\vec{w})I+i(\vec{v}\times\vec{w})\vec{\sigma}$ in the last step.

Next, we consider the product $(I\otimes H)(H\otimes I)$ and similarly obtain that, when restricted to terms of the form $A\otimes\_\otimes B$, its the Pauli expansion is given by

$$\big(I\otimes(\alpha_{IB}I+\vec{c}_B\cdot\vec{\sigma})\otimes B\big)\big(A\otimes(\alpha_{AI}I+\vec{r}_A\cdot\vec{\sigma})\otimes I\big) = \tag{5}$$

$$A\otimes\big((\alpha_{AI}\alpha_{IB}+\vec{c}_B\cdot\vec{r}_A)I+(\alpha_{AI}\vec{c}_B+\alpha_{IB}\vec{r}_A+i(\vec{c}_B\times\vec{r}_A))\cdot\vec{\sigma}\big)\otimes B. \tag{6}$$

Since the coefficients in the Pauli expansions of $(H \otimes I)(I \otimes H)$ have to coincide with those in the expansion of $(I \otimes H)(H \otimes I)$, we know that the difference between expressions (4) and (6) equals zero. Considering the middle tensor and canceling some therms gives

$$(\vec{r}_A \times \vec{c}_B) \cdot \vec{\sigma} = (\vec{c}_B \times \vec{r}_A) \cdot \vec{\sigma}. \tag{7}$$

Since $\vec{v} \times \vec{w} = -\vec{w} \times \vec{v}$, we obtain that $\vec{r}_A \times \vec{c}_B = 0$. This further implies that $\vec{r}_A$ and $\vec{c}_B$ are collinear, that is, $\dim(\mathrm{span}\{\vec{r}_A, \vec{c}_B\}) \leq 1$. Since we can choose arbitrary $A, B \in \{I, X, Y, Z\}$, it must be that all the vectors $\vec{r}_A$ and $\vec{c}_B$ must lie in the same one-dimensional subspace, i.e.,

$$\dim\left(\mathrm{span}\left\{\vec{r}_A, \vec{c}_B : A, B \in \{I, X, Y, Z\}\right\}\right) \leq 1. \tag{8}$$

Let us now consider a 3×3 matrix M whose rows and columns are indexed by Pauli matrices X, Y and Z and its entries are defined via $M_{AB} = \alpha_{AB}$. Then the vectors $\vec{c}_A$ are the columns of M and $\vec{r}_B$ are its rows. From Equation (8), we see that M has rank at most one. Moreover, the row and column spaces of M must coincide as

$$\mathrm{span}\left(\{\vec{r}_X, \vec{r}_Y, \vec{r}_Z\}\right) = \mathrm{span}\left(\{\vec{c}_X, \vec{c}_Y, \vec{c}_Z\}\right). \tag{9}$$

These two observations imply that $M = \vec{v}\vec{v}^T$ for some $\vec{v} \in \mathbb{R}^3$. So we can express our Hamiltonian H as

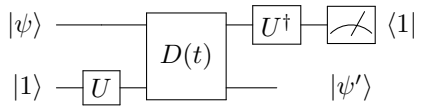
$$H = \alpha_{II} I \otimes I + (a\vec{v} \cdot \vec{\sigma}) \otimes I + I \otimes (b\vec{v} \cdot \vec{\sigma}) + (\vec{v} \cdot \vec{\sigma}) \otimes (\vec{v} \cdot \vec{\sigma}), \tag{10}$$

where $a, b \in \mathbb{R}$ are such that $\vec{r_I} = a\vec{v}$ and $\vec{c_I} = b\vec{v}$. If we pick a unitary U that diagonalizes $\vec{v} \cdot \vec{\sigma}$, then from Equation (10) we see that $U \otimes U$ diagonalizes our Hamiltonian H. This concludes the proof. ◀

B Inverting L matrices using postselection gadgets

We now prove Claim 3.2.

Proof. We will need two additional gadgets for our construction. First, consider a modification of the gadget for $L(t)$, where we start the qubit in the $|1\rangle$ state and postselect on the $|1\rangle$ state:



By a direct calculation, one can show the linear transformation performed on $|\psi\rangle$ is given by

$$M(t) = \frac{1}{|\alpha||\beta|\sqrt{e^{-2it} - e^{2it}}} \begin{pmatrix} |\beta|^2 e^{ia't} & -\alpha\beta^* e^{it} \\ -\alpha^*\beta e^{i} & |\alpha|^2 e^{id't} \end{pmatrix}.$$

This is tantalizingly close to the inverse of L, which is

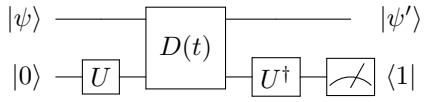
$$L(t)^{-1} = \frac{1}{|\alpha||\beta|\sqrt{e^{-2it} - e^{2it}}} \begin{pmatrix} |\beta|^2 e^{id't} & -\alpha\beta^* e^{it} \\ -\alpha^*\beta e^{it} & |\alpha|^2 e^{ia't} \end{pmatrix}.$$

The only thing that is off is that the phase of the upper left and bottom right entries are incorrect. We now break into three cases to describe how to correct the phases in each. (Recall that $d' = -2 - a'$ as our without loss of generality our Hamiltonian is traceless).

Case 1: $a' = d' = -1$. In this case we already have $M(t) = L^{-1}(t)$, so we have found the inverse.

Case 2: $a' = 1, d' = -3$ OR $a' = -3, d' = 1$. We will prove the case $a' = 1$; an analogous proof holds for $a' = -3$.

To correct the phases in $M(t)$, we need to introduce an additional gadget:



In other words, instead of using the gate in a teleportation-like protocol, we instead use it to apply phases to $|\psi'\rangle$. This gate performs the following transformation on the input state:

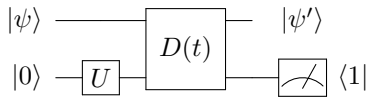
$$N(t) = \frac{1}{\sqrt{(e^{it} - e^{ia't})(e^{id't} - e^{it})}} \begin{pmatrix} e^{it} - e^{ia't} & 0 \\ 0 & e^{id't} - e^{it} \end{pmatrix}.$$

In the case that $a' = 1$, this gadget becomes singular, and hence it performs the operation

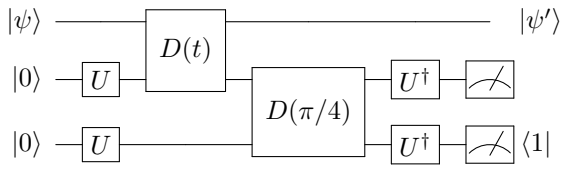
$$\begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$$

In other words, this gadget postselects the qubit involved on the state $|1\rangle$. This holds in particular for $t = \pi/4$. (In fact it holds for any t such that $e^{-3it} \neq e^{it}$, in which case it becomes undefined).

By composing $N(\pi/4)$ with other gadgets, this now empowers us to create gadgets in which we postselect on $|1\rangle$ on lines which do not end in $U^\dagger$. For instance, we can create the following gadget:



Which one can easily check is equivalent to the following circuit, which maintains the property that every line begins and ends with U and $U^\dagger$.



This is simply composing the gadget with $N(\pi/4)$. (Here the output of the middle qubit is an independent sample from measuring the state $U^\dagger|1\rangle$ in the computational basis).

This gadget performs the following operation on $|\psi\rangle$:

$$P(t) \propto \begin{pmatrix} e^{it} & 0 \\ 0 & e^{-3it} \end{pmatrix} \propto \begin{pmatrix} e^{2it} & 0 \\ 0 & e^{-2it} \end{pmatrix}$$

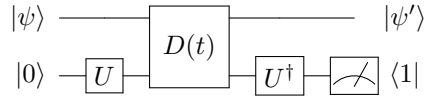
In other words, the matrix $P(t)$ is a phase gate by phase $\theta = 2t$.

The construction of arbitrary phase gates suffices to correct the diagonal phases of $M(t)$, because for any matrix $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$ we have that

$$\begin{pmatrix} e^{i\theta/2} & 0 \\ 0 & e^{-i\theta/2} \end{pmatrix} \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} e^{i\theta/2} & 0 \\ 0 & e^{-i\theta/2} \end{pmatrix} = \begin{pmatrix} ae^{i\theta} & b \\ c & de^{-i\theta} \end{pmatrix}$$

Hence by choosing $\theta = (d' - a')t$, and multiplying $M(t)$ by this matrix on both sides, we obtain $L^{-1}(t)$ as desired. Clearly this construction is efficient, i.e. the postselection gadget is of constant size, and one can efficiently compute the times to run the Hamiltonians in the gadget to high precision. This completes the proof.

Case 3: $a' \neq \pm 1, -3$. To correct the phases in $M(t)$, we need to consider the same gadget $N(t)$ which we used in Case 2:



In other words, instead of using the gate in a teleportation-like protocol, we instead use it to apply phases to $|\psi'\rangle$. This gate performs the following transformation on the input state:

$$N(t) = \frac{1}{\sqrt{(e^{it} - e^{ia't})(e^{id't} - e^{it})}} \begin{pmatrix} e^{it} - e^{ia't} & 0 \\ 0 & e^{id't} - e^{it} \end{pmatrix}$$

Since N is a diagonal matrix, the only physical quantity that matters is the ratio $r(t)$ of its two entries, which is a complex number given by

$$r(t) = \frac{e^{it} - e^{ia't}}{e^{id't} - e^{it}}.$$

If $r(t)$ takes on a certain value, then it immediately follows that $N(t) = \pm \begin{pmatrix} \sqrt{r} & 0 \\ 0 & \sqrt{r}^{-1} \end{pmatrix}$, because of our normalization. Furthermore, if we compose $N(s)N(t)$, then the ratio of the resulting diagonal matrix is $r(s)r(t)$. Note the ± 1 term is an irrelevant global phase, so we omit it in the further calculations.

We will now show that for any complex phase $e^{i\theta}$, where $\theta \neq 0, \pi$, there exists a finite set of times $t_1, t_2, ...t_k$, $s_1, s_2, ...s_{k'}$ such that

$$N(t_1)N(t_2)...N(t_k)N(s_1)N(s_2)...N(s_{k'}) = \begin{pmatrix} e^{i\theta/2} & 0 \\ 0 & e^{-i\theta/2} \end{pmatrix}$$

As previously mentioned in Case 2, the construction of such matrices suffices to correct the diagonal phases of $M(t)$, because for any matrix $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$ we have that

$$\begin{pmatrix} e^{i\theta/2} & 0 \\ 0 & e^{-i\theta/2} \end{pmatrix} \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} e^{i\theta/2} & 0 \\ 0 & e^{-i\theta/2} \end{pmatrix} = \begin{pmatrix} ae^{i\theta} & b \\ c & de^{-i\theta} \end{pmatrix}$$

Hence by choosing $\theta = (d' - a')t$, and multiplying $M(t)$ by this matrix on both sides, we obtain $L^{-1}(t)$ as desired and this will complete the proof.

To prove this, we will prove two separate facts. First, we will show that given θ, there exists a sequence $t_1, t_2, ...t_k$ such that $N(t_1)N(t_2)...N(t_k) = \begin{pmatrix} ce^{i\theta/2} & 0 \\ 0 & \frac{1}{c}e^{-i\theta/2} \end{pmatrix}$ for some $c \in \mathbb{R}^+$. Next, we will show that for any $c \in \mathbb{R}$, there exists a sequence $s_1, s_2, ...s_{k'}$ of times such that $N(s_1)N(s_2)...N(s_{k'}) = \begin{pmatrix} 1/c & 0 \\ 0 & c \end{pmatrix}$. Together these imply the claim.

Moreover, we will show this construction is efficiently computable. More specifically, suppose you want to find invert L. The for each L the size of the postselection gadget required to invert L is of constant size. Additionally, the amount of computational time

required to compute the values of t_i and s_i to ensure that we find L^{-1} to accuracy ϵ scales as polylog$(k_L * 1/\epsilon)$, where k_L is a constant which depends on L. Furthermore, the times t_i and s_i are upper bounded by a constant which only depends on the value of a'. For convenience we will refer to these properties as "the construction is efficiently computable."

At first glance it might sound like this definition of "efficiently computable" is too weak, because the inverses of arbitrary L matrices might require large postselection gadgets, or might require a long time to compute the values of the t_i and s_i to sufficient accuracy. However, later in our construction we will use the fact that for any fixed Hamiltonian H, we will only need to invert a fixed number of L matrices. Hence for fixed H, the size of the postselection gadgets which appear in our circuit will be upper bounded by a constant depending on H only, but not on the size of the problem we are solving under postselection. Furthermore, for fixed H, we can compute the times t_i, s_i required to invert the relevant L matrices to exponential accuracy in polylog$(1/epsilon)$ time (where a hidden constant k_L depending on L has been absorbed into the big-O notation).

▶ **Claim B.1.** *For any $\theta \in (0, 2\pi)$, there exists a sequence $t_1, t_2, ...t_k$ such that*

$$N(t_1)N(t_2)...N(t_k) = \begin{pmatrix} ce^{i\theta/2} & 0 \\ 0 & e^{-u\theta/2}/c \end{pmatrix}$$

for some $c \in \mathbb{R}^+$. Furthermore, this construction is computationally efficient.

Proof. To see this, consider the expression for the ratio

$$r(t) = \frac{e^{it} - e^{ia't}}{e^{id't} - e^{it}} = -\frac{1 - e^{i(a'-1)t}}{1 - e^{i(d'-1)t}}.$$

Let Phase(c) denote the phase of c modulo 2π. Then by direct calculation we have that

$$\begin{aligned}
\text{Phase}(r(t)) &= \pi + \text{Phase}\left(\frac{1 - e^{i(a'-1)t}}{1 - e^{i(-3-a')t}}\right) \\
&= \pi + \text{Phase}\left(1 - e^{i(a'-1)t}\right) - \text{Phase}\left(1 - e^{i(-3-a')t}\right) \\
&= \pi + \text{Phase}\left(1 - e^{i(a'-1)t}\right) + \text{Phase}\left(1 - e^{i(3+a')t}\right) \\
&= \left(\pi + \left(\frac{(a'-1)t}{2} \mod \pi\right) + \left(\frac{(3+a')t}{2} \mod \pi\right)\right) \mod 2\pi \\
&= (\pi + (t' \mod \pi) + (Rt' \mod \pi)) \mod 2\pi
\end{aligned}$$

Where $t' = (a'-1)t/2$ and $R = \frac{(3+a')}{(1-a')}$. Since we are in the case that $a' \neq \pm 1, -3$, we are promised that R is well-defined and $R \neq 0, 1$. Also note that we cannot have that $R = -1$ because this would imply $3 = -1$, a contradiction.

Suppose $R > 0$ (an analogous proof holds for $R < 0$). Then for $t' \in [0, \min(\pi, \pi/R)]$, we know that Phase$(r(t')) = \pi + (R+1)t'$, because in this range t' is sufficiently small such that both $t' \mod \pi = t'$ and $Rt' \mod \pi = Rt'$. Hence using t' in this interval, we can achieve any phase in $(\pi, \pi + s)$ where $s = (R+1)\min(\pi, \pi/R)$. For any $R \neq 0, -1$ this range is of constant size. Thus by multiplying together $1/s$ phases in the range $(\pi, \pi + s)$, one can achieve any phase in $(0, 2\pi)$, as desired.

Note that this construction is manifestly efficient; the t_i's are upper bounded by a constant $\min(\pi, \pi/R)$ which is a function of H only, and computing them to polynomially many digits requires polynomial time, as it just requires simple addition. ◀

▶ **Claim B.2.** *For any $c \in \mathbb{R}^+ - \{1\}$, there exists a finite sequence $s_1, s_2, ...s_k$ such that*

$$N(s_1)N(s_2)...N(s_k) = \begin{pmatrix} 1/c & 0 \\ 0 & c \end{pmatrix}.$$

Proof. Consider products of matrices of the form $N(s)N(-s)$ for $s \in \mathbb{R}+$. Let $f(s) = r(s)r(-s)$. One can check by direct calculation that

$$f(s) = \frac{1 - \cos((1-a')s)}{1 - \cos((3+a')s)}$$

In other words, the product of the ratios is real and positive, hence the resulting matrix $N(s)N(-s)$ is of the form $\begin{pmatrix} 1/\ell & 0 \\ 0 & \ell \end{pmatrix}$ for some $\ell \in \mathbb{R}^+$. Note since we are in the case $a' \neq \pm 1, -3$ this ratio is well-defined.

If we redefine $s' = s/(1-a')$, and set $R = (1-a')/(3+a')$, then this ratio becomes

$$\frac{1 - \cos s'}{1 - \cos Rs'}.$$

We know $R \neq 0, 1$ because we have $a' \neq \pm 1, 3$, and furthermore $R \neq -1$ as well, since this would imply $1 = -3$, a contradiction.

For clarity of explanation assume $R > 0$; an analogous proof holds for the case $R < 0$.

Next we claim that the range of $f(s)$ as s varies over R includes the interval

$$(\min(R^{-2}, R^2), \max(R^{-2}, R^2)).$$

Since $R \neq 1$ this is an interval of constant size around 1. To see this, we will break into two cases.

First, assume $R > 1$. Consider the value of this function when $s' \in (0, \pi/R)$. The function $f(s')$ in continuous in this range. Additionally $\lim_{s' \to 0} f(s') = 1/R^2$ by L'Hôpital's rule, and $\lim_{s' \to \pi/R} = +\infty$. Hence the range of f covers $(R^{-2}, +\infty) = (\min(R^{-2}, R^2), +\infty)$ by the mean value theorem.

Next, assume $0 < R < 1$. Now consider the value of the function when $s' \in (0, \pi)$. Again the function is continuous in this range, and we have $\lim_{s' \to 0} f(s') = 1/R^2$ by L'Hôpital's rule, and $\lim_{s' \to \pi} = 0$. Hence the range of f covers $(0, R^{-2}) = (0, \max(R^{-2}, R^2))$ by the mean value theorem.

Hence in either case, by choosing an appropriate value of s', we can set $f(s)$ to be any real value in a finite-length interval containing 1. Hence for any target ratio $c^2 \in \mathbb{R}^+$, one can take a finite product of $O(\log(c))$ values of $f(s)$ such that $f(s_1)f(s_2)...f(s_k) = c^2$. This implies the claim.

Note that this construction is efficient. First, the times s_i are upper bounded by $\min(\pi, \pi/R)$, which is a constant which depends on the Hamiltonian H only. Second, to compute each individual time s_i, one simply needs to solve the problem

$$\frac{1 - \cos s'}{1 - \cos Rs'} = k$$

For some $k \in (\min(R^{-2}, R^2), \max(R^{-2}, R^2))$ and s' in $(0, \min(\pi, \pi/R))$. In the region of s where the value of this function is between $\min(R^{-2}, R^2)$ and $\max(R^{-2}, R^2))$, the derivatives of this function are bounded by a function of R only. Furthermore, the derivatives of these terms are computable to accuracy ϵ in time $\text{polylog}(1/\epsilon)$ time using the Taylor series for sine and cosine. Hence Newton's method can be used to solve this problem, and will achieve

quadratic convergence, i.e. for each step you run Newton's method, the error is squared, and the number of digits of accuracy achieved doubles. Hence one can compute each time t_i to accuracy ϵ in polylog$(1/\epsilon)$ time as desired. Furthermore, since inverting any particular L only requires inverting some fixed $c \in \mathbb{R}^+$ using Claim B.2, an error ϵ in an individual $N(s_i)$ matrices contributes $c\epsilon$ error to the operator norm[8] of $N(s_1)...N(s_k)$, and hence $c\epsilon$ error to the operator norm of L^{-1}. Hence this construction is "computationally efficient" for each fixed L as defined previously. ◀

This completes the proof in Case 3 and hence the entire proof. ◀

C Showing density in $SL(2, \mathbb{C})$

We now prove Claim 3.3.

Proof of Claim 3.3. To show that $S = SL(2, \mathbb{C})$, we will first show that S is a group, and then show S is a Lie group.

▶ **Claim C.1.** *S is a group.*

Proof. Clearly, if we only took finite products of these elements, the resulting set of matrices would be a group, because we have the inverses of every element in the generating set. So what we need to show is that taking the closure of this set of matrices still yields a group. To see this, suppose that some element $s \in S \subseteq SL(2, \mathbb{C})$ is the limit of a sequence $L_1, L_2, \ldots$ where each L_i is a finite product of element of the form $L(D(t_1, t_2))$, and $\lim_{i\to\infty} L_i = s$. Now consider the sequence $L_1^{-1}, L_2^{-1}, \ldots$. We claim that $\lim_{i\to\infty} L_i^{-1} = s^{-1}$. To see this, simply note that for a 2×2 matrix $\left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right) \in SL(2, \mathbb{C})$, its inverse is given by $\left(\begin{smallmatrix} d & -b \\ -c & a \end{smallmatrix}\right)$. Since the limit point s exists in $SL(2, \mathbb{C})$, the limit of each matrix entry of the L_i's must converge as well to the entries of s. Hence the entries of the sequence L_i^{-1} converges to the entries of s^{-1}. ◀

Note that it is critical that we've taken the closure in $SL(2, \mathbb{C})$; if we took the closure in the set of 2×2 complex matrices, this would not necessarily be true.

We have now established that S is a group. Furthermore, S is a closed subgroup of $SL(2, \mathbb{C})$ by construction, and $SL(2, \mathbb{C})$ is a Lie group. We now invoke a well-known theorem from Lie theory.

▶ **Theorem C.2** (Cartan's Theorem [12] or the Closed Subgroup Theorem)**.** *Any closed subgroup of a Lie group is a Lie group.*

▶ **Corollary C.3.** *S is a Lie group.*

Now that we know S is a Lie group, we can use facts from Lie theory to show $S = SL(2, \mathbb{C})$. We will summarize the basics here, but a more complete treatment can be found in e.g. [21] or a more advanced textbook on Lie groups.

A Lie group is a continuous manifold which is also a group, for which the group operations are smooth. In this work we will only consider matrix groups, i.e. continuous groups of complex matrices. For any Lie group G, one can define the Lie algebra of G, denoted Lie(G), to be the tangent space to the group G at the identity. More concretely, suppose that you

[8] This is because for non-unitary matrices, the norm of the singular values are not one. Hence when considering the product AB, where λ_{max} is the largest singular value of A, an ϵ error in B will induce an $\lambda_{max}\epsilon$ error in AB.

have a smooth path $\gamma(t) : \mathbb{R} \to G \subseteq GL(n, \mathbb{C})$ in G, such that $\gamma(0) = I$. Then the matrix $\frac{\partial}{\partial t}\gamma(t)\Big|_{t=0}$ belongs to the tangent space of G at the identity. One can show that $\mathrm{Lie}(G)$ obeys the following properties [21]:

1. $\mathfrak{g}$ is a real vector space, i.e. $g_1, g_2 \in \mathfrak{g} \Rightarrow ag_2 + bg_2 \in \mathfrak{g}$ for any $a, b \in \mathbb{R}$.
2. $\mathfrak{g}$ is closed under commutators, i.e. $g_1, g_2 \in \mathfrak{g} \Rightarrow [g_1, g_2] \triangleq g_1g_2 - g_2g_1 \in \mathfrak{g}$ for any $a, b \in \mathbb{R}$.
3. Let $\exp(A) = I + A + \frac{A^2}{2} + \frac{A^3}{6} + \ldots + \frac{A^n}{n!} + \ldots$. Then we have that for all $g \in \mathfrak{g}$, $\exp(g) \in G$. In other words, the function exp maps from the Lie algebra into the Lie group.
4. $\mathfrak{g}$ is closed under taking commutators with the group G. That is, for any $G_1 \in G$ and $g \in \mathfrak{g}$, we have $G_1 g G_1^{-1} \in \mathfrak{g}$.

To show that $S = SL(2, \mathbb{C})$, we will consider $\mathfrak{g} \triangleq \mathrm{Lie}(S)$. We will then show that $\mathfrak{g} = \mathfrak{sl}(2, \mathbb{C})$, which is the Lie algebra of $SL(2, \mathbb{C})$, which consists of all traceless two by two complex matrices. By property 3, this implies that $\exp(\mathfrak{sl}(2, \mathbb{C})) \subseteq S$. From this, we will leverage the following fact:

▶ **Claim C.4.** $\exp(\mathfrak{sl}(2, \mathbb{C}))$ *is dense in* $SL(2, \mathbb{C})$.

Proof. It is well known [21] that $\exp(\mathfrak{sl}(2, \mathbb{C}))$ contains all matrices in $SL(2, \mathbb{C})$ except matrices A for which $\mathrm{Tr}(A) = -2$ and $A \neq -I$. This implies the claim. ◀

Hence to prove Claim 3.3, it suffices to prove the following claim:

▶ **Claim C.5.** $\mathfrak{g} \triangleq \mathrm{Lie}(S)$ *spans* $\mathfrak{sl}(2, \mathbb{C})$*, i.e. all* 2×2 *traceless matrices.*

Proof. Consider elements of the form

$$M(t, s) \triangleq L(t)L(s)^{-1}.$$

As t, s vary over $(0, \pi) \cup (\pi, 2\pi)$, these form continuous paths within S. In particular, at the point where $s = t$, this path passes through the identity. Now consider

$$g(v) \triangleq \frac{\partial}{\partial t}\left[M(t, s)\right]\Big|_{s=t=v}.$$

These are tangent vectors to paths in S, evaluated as they pass through the identity. Hence we have that $g(v) \in \mathfrak{g}$ for all $v \in (0, \pi) \cup (\pi, 2\pi)$. By direct calculation, one can show that

$$g(v) = -\frac{1}{2\sin(2v)} \begin{pmatrix} (a'+1)e^{-2iv} & \frac{\alpha}{\beta}(1-a')e^{i(1+a')v} \\ \frac{\beta}{\alpha}(3+a')e^{i(-1-a')v} & -(a'+1)e^{-2iv} \end{pmatrix}$$

where we have simplified using the fact that $d' = -2 - a'$.

We will now break into cases to show that these matrices span the entire Lie algebra. We begin with the generic case and then give the special cases. In the special cases, we will also add additional postselection gadgets to our model in order to get single-qubit transformations which span all traceless matrices. The gadgets introduced are inherently closed under taking inverses. So this simply reflects that for very particular Hamiltonians, our L matrices need additional help to span all 1-qubit operations.

Case 1: $a' \neq \pm 1, -3$. In this case all of the entries of $g(v)$ are non-zero.

$$g(v) = -\frac{1}{2\sin(2v)} \begin{pmatrix} (a'+1)e^{-2iv} & \frac{\alpha}{\beta}(1-a')e^{i(1+a')v} \\ \frac{\beta}{\alpha}(3+a')e^{i(-1-a')v} & -(a'+1)e^{-2iv} \end{pmatrix}$$

We can therefore rewrite $g(v)$ with four non-zero parameters $k_1 \in \mathbb{R}$, $k_2, k_3 \in \mathbb{C}$, and using a new parameter $v' = -2v$:

$$g(v) \propto \begin{pmatrix} e^{v'} & k_2 e^{ik_1 v'} \\ k_3 e^{-ik_1 v'} & -e^{iv'} \end{pmatrix}$$

Here we omit real coefficients as the Lie algebra is closed under scalar multiplication by $\mathbb{R}$. The fact that $a' \neq \pm 1, -3$ also implies that $k_4 \neq \pm 1$

Now consider the value of $g(v')$ for small values of v'. In particular, pick a $\theta << 1$. Then we have that

$$g(\pm\theta) \propto \begin{pmatrix} (A \pm Bi) & k_2(C \pm Di) \\ k_3(C \mp Di) & -(A \pm Bi) \end{pmatrix}$$

for some nonzero real coefficients $A, B, C, D \in \mathbb{R}$. Taking the sum and difference of these matrices, we see the following are elements of the Lie algebra:

$$\begin{pmatrix} A & k_2 C \\ k_3 C & -A \end{pmatrix} \quad \begin{pmatrix} Bi & k_2 Di \\ -k_3 Di & -Bi \end{pmatrix}$$

Likewise, by considering taking the sum and difference of $g(\pm 2\theta)$, we get there exist nonzero $A', B', C', D' \in \mathbb{R}$ such that the lie algebra contains.

$$\begin{pmatrix} A' & k_2 C' \\ k_3 C' & -A' \end{pmatrix} \quad \begin{pmatrix} B'i & k_2 D'i \\ -k_3 D'i & -B'i \end{pmatrix}$$

Furthermore, since sine and cosine are nonlinear, and $k_1 \neq 0, \pm 1$, the vectors (A, C) and (A', C') are linearly independent. Likewise the vectors (B, D) and (B', D') are linearly independent. Hence by taking linear combinations of these matrices, we have that any matrix of the form

$$\begin{pmatrix} E & k_2 F \\ k_3 F^* & -E \end{pmatrix}$$

is in the Lie algebra for any $E, F \in \mathbb{C}$. Hence our Lie algebra spans at least these two complex dimensions. Now we take the closure of such matrices under commutators. Suppose $A, B, C, D \in \mathbb{C}$. We have that

$$\left[\begin{pmatrix} A & k_2 B \\ k_3 B^* & -A \end{pmatrix}, \begin{pmatrix} C & k_2 D \\ k_3 D^* & C \end{pmatrix} \right] = \begin{pmatrix} k_2 k_3 (BD^* - B^* D) & 2k_2(AD - BC) \\ 2k_3(B^* C - AD^*) & k_2 k_3 (B^* D - BD^*) \end{pmatrix}$$

Since we previously showed all traceless diagonal matrices are in the Lie algebra, this implies the following matrices are in the Lie algebra:

$$\begin{pmatrix} 0 & 2k_2(AD - BC) \\ 2k_3(B^* C - AD^*) & 0 \end{pmatrix}$$

By setting A, D, B, C such that $(AD - BC)^* \neq (B^* C - AD^*)$, we can see that these matrices span the remaining two real dimensional space of off-diagonal matrices. Hence our Lie algebra spans all traceless matrices. This completes the proof in Case 1.

Case 2: $a' = 1$ or $a' = -3$. We will prove the claim for $a' = 1$; an analogous proof holds for $a' = -3$. (These are the Hamiltonians $\mathrm{diag}(1,1,1,-3)$ and $\mathrm{diag}(-3,1,1,1)$, which are identical except the role of 0 and 1 is switched.).

In this case we have that

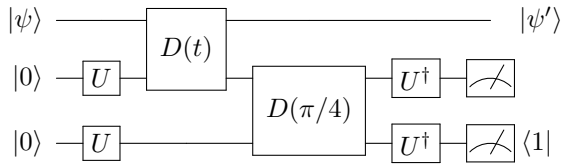
$$g(v) \propto \begin{pmatrix} e^{-2iv} & 0 \\ 2\frac{\beta}{\alpha}e^{-2iv} & -e^{-2iv} \end{pmatrix}$$

By evaluating $g(v)$ at $\pm\theta$ and $\pm 2\theta$ for some small value of θ, by the same arguments put forth in Case 1, these matrices span the space of matrices of the form

$$\begin{pmatrix} A+Bi & 0 \\ 2\frac{\beta}{\alpha}(A+Bi) & -A-Bi \end{pmatrix}$$

Where $A, B \in \mathbb{R}$ are arbitrary real parameters.

We will now use another postselection gadget, which is inherently closed under taking inverses, to boost the span of the algebra to all of $\mathfrak{sl}(2,\mathbb{C})$. This is the same gadget which appears in the construction of L^{-1} in Appendix B.



This gadget performs the operation

$$P(t) \propto \begin{pmatrix} e^{it} & 0 \\ 0 & e^{-3it} \end{pmatrix} \propto \begin{pmatrix} e^{2it} & 0 \\ 0 & e^{-2it} \end{pmatrix}$$

Hence its Lie algebra spans the space of traceless diagonal imaginary matrices. Combining this with the previous result, we see the Lie algebra now spans the space

$$\begin{pmatrix} A+Bi & 0 \\ 2\frac{\beta}{\alpha}(A+Ci) & -A-Bi \end{pmatrix}$$

Where $A, B, C \in \mathbb{R}$ are arbitrary real parameters.

Now consider taking commutators of such matrices; one can easily see that for $A, B, C, D, E, F \in \mathbb{R}$,

$$\left[\begin{pmatrix} A+Bi & 0 \\ 2\frac{\beta}{\alpha}(A+Ci) & -A-Bi \end{pmatrix}, \begin{pmatrix} D+Ei & 0 \\ 2\frac{\beta}{\alpha}(D+Fi) & -D-Ei \end{pmatrix}\right] = \begin{pmatrix} 0 & 0 \\ 4\frac{\beta}{\alpha}(A+Ci)(D+Ei) & 0 \end{pmatrix}$$

Hence by appropriate choice of A, C, D, E these commutators span all complex values in the lower left hand corner. So our Lie algebra now spans

$$\begin{pmatrix} A+Bi & 0 \\ C+Di & -A-Bi \end{pmatrix}$$

Where $A, B, C, D \in \mathbb{R}$ are arbitrary real parameters. In other words we span all traceless lower triangular matrices.

Next we will use the fact that the Lie algebra is closed under conjugation by the group. Therefore it must contain all elements of the form

$$L(t)\begin{pmatrix} A & 0 \\ B & -A \end{pmatrix}L^{-1}(t)$$

where A, B are now complex parameters

Since we already span lower triangular matrices, the only relevant entry of the above matrix is the upper-right entry, as we can zero out the other entries by adding lower triangular matrices. This upper left entry is proportional to

$$i\left(-2\alpha\beta^*|\alpha|^2e^{2it}A - \alpha^2\beta^{*2}e^{2it}B\right)$$

Since α and β are non-zero, and setting $B = 0$, we can see that by choosing A we can set this value to be any complex number. Hence our Lie algebra must span

$$L(t)\begin{pmatrix} A & C \\ B & -A \end{pmatrix}L^{-1}(t)$$

Where $A, B, C \in \mathbb{C}$, that is all of $\mathfrak{sl}(2,\mathbb{C})$, as desired. This completes the proof of Claim 2.

Case 3: $a' = -1$. In this case we have that

$$g(v) = -\frac{1}{\sin(2v)}\begin{pmatrix} 0 & \frac{\alpha}{\beta} \\ \frac{\beta}{\alpha} & 0 \end{pmatrix}$$

Thus the matrices $g(v)$ span a one-dimensional space. Since the Lie algebra is closed under scalar multiplication by reals, the factor of $\frac{-1}{\sin(2v)}$ out front is irrelevant, and we will drop real prefactors in future calculations.

We will now use the fact the Lie algebra is closed under conjugation by the group. Consider matrices of the form

$$T(s,v) = L(s)g(v)L(s)^{-1} \propto i\begin{pmatrix} |\beta|^4 - |\alpha|^4 & |\alpha|^4\frac{\alpha}{\beta}e^{-2is} - \alpha\beta^*|\beta|^2e^{2is} \\ \frac{\beta}{\alpha}|\beta|^4e^{-2is} - \alpha^*\beta|\alpha|^2e^{2is} & |\alpha|^4 - |\beta|^4 \end{pmatrix}$$

where the proportionality is over real scalar multiples. Here we have simplified using the fact we are in the case $a' = d' = -1$. This is well defined for any s and v which are not integer multiples of π.

Now we break into two subcases:

Subcase A: $|\alpha|^2 \neq |\beta|^2$. In this case, the matrix $T(s,v)$ has a nonzero entry on the diagonals. Hence the matrix $T(s,v)$ has the form

$$T(s,v) \propto i\begin{pmatrix} k_1 & k_2e^{-2is} - k_3e^{2is} \\ k_4e^{-2is} - k_5e^{2is} & -k_1 \end{pmatrix}$$

Where $k_1 \in \mathbb{R}$ is nonzero, $k_2, k_3, k_4, k_5 \in \mathbb{C}$ are nonzero. One can easily check that the constraint $|\alpha|^2 \neq |\beta|^2$ further implies that k_2, k_3, k_4, k_5 have four distinct values, i.e. $k_i \neq k_j$ for any $i \neq j$, $i, j \geq 2$. For instance, to see that $k_2 \neq k_3$, note that if $k_2 = k_3$ then $|\alpha|^4\frac{\alpha}{\beta} = \alpha\beta^*|\beta|^2$, which implies $|\alpha|^4 = |\beta|^4$, a contradiction.

Furthermore, one can show that there cannot exist a constant[9] K such that $k_2 = Kk_4$ and $k_3 = Kk_5$, because this would imply $|K| = |\frac{\alpha}{\beta}|^6 = |\frac{\alpha}{\beta}|^2$ which is a contradiction if $|\alpha| \neq |\beta|$. Hence the matrices $T(s,v)$ span matrices of the form

$$\begin{pmatrix} Ai & B + Ci \\ D + Ei & -Ai \end{pmatrix}$$

[9] If this were the case, the matrices $T(s,v)$ would only span matrices of the form $\begin{pmatrix} Ai & B+Ci \\ K(B+Ci) & -Ai \end{pmatrix}$. Fortunately this does not happen in this case.

where $A, B, C, D, E \in \mathbb{R}$ are arbitrary real parameters. Now taking the closure of such matrices under commutators, one can easily see this spans all traceless matrices. Hence the Lie algebra spans $\mathfrak{sl}(2, \mathbb{C})$ as desired.

Subcase B: $|\alpha|^2 = |\beta|^2 = 1/2$. In this case the Hamiltonians generated are of the form $X(\theta) \otimes X(\theta)$, so are not covered in the scope of this theorem. Note that the Lie alebgra of the L gadgets here only span a two dimensional subspace of the form

$$\begin{pmatrix} 0 & e^{-i\theta}(A+Bi) \\ e^{i\theta}(A+Bi) & 0 \end{pmatrix}$$

where $A, B \in \mathbb{R}$. This is closed under conjugation and does not span $\mathfrak{sl}(2, \mathbb{C})$. ◀

◀

D Proof of postselected universality when $b \neq c$

Here we consider the postselected universality of circuits with entangling Hamiltonians for which $H \neq THT$. The proof in this case will follow analogously to the main proof. Furthermore, the construction of the inverse gadgets will have a much cleaner construction than the case $H = THT$.

Suppose we have a commuting Hamiltonian H such that $H \neq THT$. By Claim 3.1, we know that $H = (U \otimes U)\,\mathrm{diag}(a, b, c, d)(U^\dagger \otimes U^\dagger)$ for some one-qubit unitary $U = \left(\begin{smallmatrix} \alpha & -\beta^* \\ \beta & \alpha^* \end{smallmatrix}\right)$ and some real parameters a, b, c, d. The trace of H contributes an irrelevant global phase to the unitary operator it generates, so without loss of generality we can assume H is traceless, i.e., $a + b + c + d = 0$. Since $H \neq THT$ we have $b \neq c$. As before, the fact H can generate entanglement starting from the computational basis implies $\alpha \neq 0, \beta \neq 0$, and $b + c \neq 0$.

Now consider the Hamiltonians

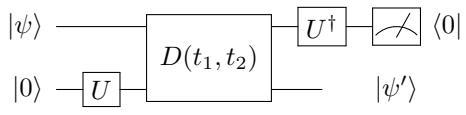
$$H_1 = \frac{1}{c^2 - b^2}(cH_{12} - bH_{21}), \quad H_2 = \frac{1}{b^2 - c^2}(bH_{12} - cH_{21})\,.$$

Since we can apply both H, $-H$, THT, and $-THT$, this allows us to apply H_1 and H_2 for independent amouts of time. Let $V(t_1, t_2)$ be the two-qubit unitary we obtain from running H_1 for time $t_1 \in \mathbb{R}$ and H_2 for time $t_2 \in \mathbb{R}$. We have

$$V(t_1, t_2) = e^{it_1 H_1} e^{it_2 H_2} = (U^{\otimes 2}) D(t_1, t_2) (U^{\dagger \otimes 2}),$$

where $D(t_1, t_2) \triangleq \mathrm{diag}(e^{ia'(t_1+t_2)}, e^{it_1}, e^{it_2}, e^{id'(t_1+t_2)})$.

Now following our previous proof, we consider the following postselection gadget:



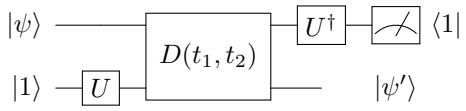
This performs the following transformation on the input state:

$$L(t_1, t_2) = \frac{1}{|\alpha||\beta|\sqrt{\left(e^{-i(t_1+t_2)} - e^{i(t_1+t_2)}\right)}} \begin{pmatrix} |\alpha|^2 e^{ia'(t_1+t_2)} & \alpha\beta^* e^{it_2} \\ \alpha^*\beta e^{it_1} & |\beta|^2 e^{id'(t_1+t_2)} \end{pmatrix}.$$

As before, this is a non-unitary transformation, and hence it is unclear how to invert L. Fortunately, when $H \neq THT$ we have the freedom to apply H_1 and H_2 for separate times, and this allows us to make a much simpler postselecting gadget to invert L, as follows:

▶ **Claim D.1.** *Given $L(t_1, t_2)$, where $t_i \in (0, \pi) \cup (\pi, 2\pi)$, it is possible to construct $L(t_1, t_2)^{-1}$ by introducing three postselections into the circuit. Furthermore, this construction is efficiently computable in the manner described above.*

Proof. We will need two additional gadgets for our construction. First, consider a modification of the gadget for $L(t_1, t_2)$, where we start the qubit in the $|1\rangle$ state and postselect on the $|1\rangle$ state:



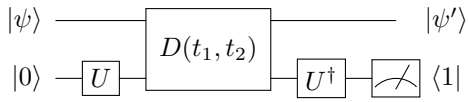
By a direct calculation, one can show the linear transformation performed on $|\psi\rangle$ is given by

$$M(t_1, t_2) = \frac{1}{|\alpha||\beta|\sqrt{\left(e^{-i(t_1+t_2)} - e^{i(t_1+t_2)}\right)}} \begin{pmatrix} |\beta|^2 e^{ia'(t_1+t_2)} & -\alpha\beta^* e^{it_2} \\ -\alpha^*\beta e^{it_1} & |\alpha|^2 e^{id'(t_1+t_2)} \end{pmatrix}$$

This is tantalizingly close to the inverse of L, which is

$$L(t_1, t_2)^{-1} = \frac{1}{|\alpha||\beta|\sqrt{\left(e^{-i(t_1+t_2)} - e^{i(t_1+t_2)}\right)}} \begin{pmatrix} |\beta|^2 e^{id'(t_1+t_2)} & -\alpha\beta^* e^{it_2} \\ -\alpha^*\beta e^{it_1} & |\alpha|^2 e^{ia'(t_1+t_2)} \end{pmatrix}$$

The only thing that is off is that the phase of the upper left and bottom right entries are incorrect. To correct these phases, we need to introduce another gadget:



In other words, instead of using the gate in a teleportation-like protocol, we instead use it to apply phases to $|\psi'\rangle$. This gate performs the following transformation on the input state:

$$N(t_1, t_2) = \frac{1}{\sqrt{(e^{it_1} - e^{ia'(t_1+t_2)})(e^{id'(t_1+t_2)} - e^{it_2})}} \begin{pmatrix} e^{it_1} - e^{ia'(t_1+t_2)} & 0 \\ 0 & e^{id'(t_1+t_2)} - e^{it_2} \end{pmatrix}$$

Since N is a diagonal matrix, the only physical quantity that matters is the ratio $r(t_1, t_2)$ of its two entries, which is a complex number given by

$$r(t_1, t_2) = \frac{e^{it_1} - e^{ia'(t_1+t_2)}}{e^{id'(t_1+t_2)} - e^{it_2}}.$$

If $r = r(t_1, t_2)$ takes on a certain value, then it immediately follows that $N(t_1, t_2) = \begin{pmatrix} \sqrt{r} & 0 \\ 0 & \sqrt{r}^{-1} \end{pmatrix}$, because of our normalization.

We will now show that by setting t_1 and t_2, we can choose $r(t_1, t_2)$ to be any complex phase $e^{i\theta}$ that we like. In fact, if $\frac{a'}{d'}$ is irrational, one can also show that one can choose t_1, t_2 to approximate any complex number; however, this will not be necessary for our construction, so we omit this here.

▶ **Claim D.2.** *For any $\theta \in (0, 2\pi)$, there exist $t_1, t_2 \in \mathbb{R}$ such that $r(t_1, t_2) = e^{i\theta}$.*

Proof. Set $t_1 = \theta$ and $t_2 = -\theta$. We immediately have

$$r(\theta, -\theta) = \frac{e^{i\theta} - 1}{1 - e^{-i\theta}} = \frac{e^{i\theta} - 1}{e^{-i\theta}(e^{i\theta} - 1)} = e^{i\theta}.$$

Note that this only works if $e^{i\theta} \neq 1$ - this is why we have omitted $\theta = 0$ from our range of θ. In other words, this gadget can be used to perform any diagonal matrix other than the identity. ◀

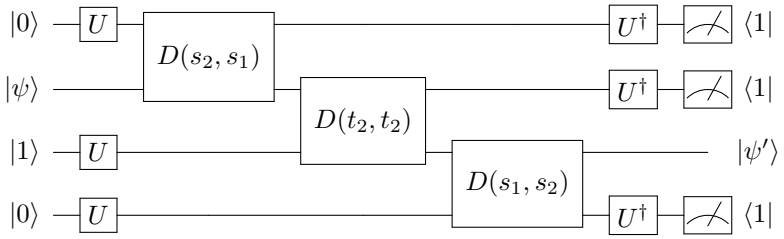
Putting this all together, we now show how to invert $L(t_1, t_2)$. Set $s_1 = i(d'(t_1 + t_2) - a'(t_1 + t_2))$ and $s_2 = -s_1$. Then we have[10]

$$N(s_1, s_2) = \begin{pmatrix} e^{\frac{i}{2}(d'(t_1+t_2)-a'(t_1+t_2))} & 0 \\ 0 & e^{-\frac{i}{2}(d'(t_1+t_2)-a'(t_1+t_2))} \end{pmatrix}$$

Now one can easily check that

$$L(t_1, t_2)^{-1} = N(s_1, s_2)M(t_1, t_2)N(s_1, s_2)\,.$$

And therefore the following gadget performs $L(t_1, t_2)^{-1}$:



(Note that s_1 and s_2 are switched in the first diagonal matrix, as we have switched the usual order of the qubits.)

Hence using these postselection gadgets, we can generate not only $L(t_1, t_2)$, but also its inverse. Furthermore, this construction is manifestly efficient, since s_1 and s_2 are efficiently computable given t_1 and t_2. ◀

We can therefore apply both $L(t_1, t_2)$ and $L(t_1, t_2)^{-1}$ in our postselected circuits. This once again allows us to apply Lie theory to determine which subset of transformations can be applied by taking products of L matrices. Following our proof of the main theorem, we now show the Lie algebra of the L matrices spans $\mathfrak{sl}(2, \mathbb{C})$. This completes the proof of postselected universality in this case in analogy with the main theorem.

▶ **Claim D.3.** *The Lie algebra of the L matrices spans $\mathfrak{sl}(2, \mathbb{C})$ in the case where $T \neq THT$.*

Proof. Consider elements of the form

$$M(t_1, t_2, s_1, s_2) \triangleq L(D(t_1, t_2))L(D(s_1, s_2))^{-1}.$$

As t_1, t_2, s_1, s_2 vary over the set

$$\{t_1, t_2 : t_1 + t_2 \in (0, \pi) \cup (\pi, 2\pi)\} \times \{s_1, s_2 : s_1 + s_2 \in (0, \pi) \cup (\pi, 2\pi)\}\,,$$

these form continuous paths within S. In particular, at the point where $s_1 = t_1$ and $s_2 = t_2$, this path passes through the identity. Now consider

$$g(v_1, v_2) \triangleq \frac{\partial}{\partial t_1}\left[M(t_1, t_2, s_1, s_2)\right]\Big|_{\substack{s_1=t_1=v_1 \\ s_2=t_2=v_2}}$$

[10] This is possible as long as $e^{i(d'(t_1+t_2)-a'(t_1+t_2))} \neq 1$. If this quantity is one, then $L(t_1, t_2)^{-1} = M(t_1, t_2)$, so no additional gadgets are necessary to obtain inverses.

and

$$h(v_1, v_2) \triangleq \frac{\partial}{\partial t_2} \left[M(t_1, t_2, s_1, s_2) \right] \Big|_{\substack{s_1 = t_1 = v_1 \\ s_2 = t_2 = v_2}}.$$

These are tangent vectors to paths in S, evaluated as they pass through the identity. Hence we have that $g(v_1, v_2)$ and $h(v_1, v_2) \in \mathfrak{g}$ for all $v_1, v_2 \in \{v_1, v_2 : v_1 + v_2 \in (0, \pi) \cup (\pi, 2\pi)\}$. By direct calculation, one can show that

$$g(v_1, v_2) = -\frac{1}{2\sin(v_1 + v_2)} \begin{pmatrix} a' e^{-i(v_1+v_2)} + \cos(v_1 + v_2) & -\frac{\alpha}{\beta} a' e^{i(a' v_1 + (a'+1) v_2)} \\ \frac{\beta}{\alpha}(2 + a') e^{i((d'+1)v_1 + d' v_2)} & -a' e^{-i(v_1+v_2)} - \cos(v_1 + v_2) \end{pmatrix}$$

and

$$h(v_1, v_2) = -\frac{1}{2\sin(v_1 + v_2)} \begin{pmatrix} a' e^{-i(v_1+v_2)} - i\sin(v_1 + v_2) & \frac{\alpha}{\beta}(1 - a') e^{i(a' v_1) + (a'+1) v_2} \\ \frac{\beta}{\alpha}(1 + a') e^{i((d'+1)v_1 + d' v_2)} & -a' e^{-i(v_1+v_2)} + i\sin(t + 1 + v_2) \end{pmatrix}$$

where we have simplified using the fact that $d' = -1 - a'$. Now suppose that we evaluate these matrices at the points where $v_1 = \theta$ and $v_2 = \frac{\pi}{2} - \theta$ for some real parameter θ; this ensures that v_1, v_2 are in the allowed set, and simplifies the above expressions to

$$\begin{aligned} g(\theta) &= -\frac{1}{2} \begin{pmatrix} -a' i & -\frac{\alpha}{\beta} a' e^{i(-\theta + (a'+1)\frac{\pi}{2})} \\ \frac{\beta}{\alpha}(2 + a') e^{i(\theta + d' \frac{\pi}{2})} & a' i \end{pmatrix} \\ &= -\frac{1}{2} \begin{pmatrix} -a' i & -\frac{\alpha}{\beta} a' e^{i\theta'} \\ \frac{\beta}{\alpha}(2 + a') e^{-i\theta'} & a' i \end{pmatrix}, \end{aligned}$$

here we define $\theta' = -\theta + (a' + 1)\frac{\pi}{2}$; this follows from the fact that $d' = -1 - a'$. Likewise, we can consider $h(v_1, v_2)$ evaluated when $v_1 = \theta$ and $v_2 = \frac{\pi}{2} - \theta$; this evaluates to

$$\begin{aligned} h(\theta) &= -\frac{1}{2} \begin{pmatrix} -i a' - i & \frac{\alpha}{\beta}(1 - a') e^{i(-\theta + (a'+1)\frac{\pi}{2})} \\ \frac{\beta}{\alpha}(1 + a') e^{i(\theta + d' \frac{\pi}{2})} & i a' + i \end{pmatrix} \\ &= -\frac{1}{2} \begin{pmatrix} -i(a' + 1) & \frac{\alpha}{\beta}(1 - a') e^{i\theta'} \\ \frac{\beta}{\alpha}(1 + a') e^{-i\theta'} & i(a' + 1) \end{pmatrix}. \end{aligned}$$

By setting the value of θ in the range $[0, 2\pi)$, we can select any values of θ' we like; hence we will work with θ' from this point forward.

For now we will assume that $a' \neq 0$ and $a' \neq 1$; we will handle the cases $a' = 0$ and $a' = -1$ separately. The proof of the general case is the most difficult one.

Case 1: $a' \neq 0$ and $a' \neq -1$. We know that $g(\theta') \in \mathfrak{g}$ and $h(\theta') \in \mathfrak{g}$. Furthermore, since $\mathfrak{g}$ is a real Lie algebra, it is closed as a vector space over $\mathbb{R}$. Hence we must also have that

$$\begin{aligned} j(\theta_1, \theta_2) &\triangleq -2 \left(\frac{1}{a' + 1} h(\theta_2) - \frac{1}{a'} g(\theta_1) \right) \\ &= \begin{pmatrix} 0 & \frac{\alpha}{\beta} \left(\frac{1 - a'}{1 + a'} e^{i\theta_2} + e^{i\theta_1} \right) \\ \frac{\beta}{\alpha} \left(e^{-i\theta_2} - \frac{2 + a'}{a'} e^{-i\theta_1} \right) & 0 \end{pmatrix} \in \mathfrak{g} \end{aligned}$$

Where we have used the assumption that $a' \neq 0$ and $a' \neq -1$. We will now show that as we vary θ_1 and θ_2, these elements $j(\theta_1, \theta_2)$ span all two by two matrices of the form $\left(\begin{smallmatrix} 0 & c_1 \\ c_2 & 0 \end{smallmatrix}\right)$, where $c_1, c_2 \in \mathbb{C}$.

To prove this, we will break into two subcases. For convenience, define

$$k = \frac{a' - 1}{a' + 1}.$$

Subcase A: $a' > 0$, i.e., $-1 < k < 1$. In this subcase, consider the matrices

$$\frac{-a'(1+a')}{4}\left[j\left(\arcsin k, \frac{\pi}{2}\right) + j\left(\pi - \arcsin k, \frac{\pi}{2}\right)\right] = \begin{pmatrix} 0 & 0 \\ \frac{\beta}{\alpha} i & 0 \end{pmatrix} \tag{11}$$

$$\frac{1+a'}{4\sqrt{a'}}\left[j\left(\arcsin k, \frac{\pi}{2}\right) - j\left(\pi - \arcsin k, \frac{\pi}{2}\right)\right] = \begin{pmatrix} 0 & \frac{\alpha}{\beta} \\ -\frac{\beta}{\alpha}\frac{2+a'}{a'} & 0 \end{pmatrix} \tag{12}$$

and

$$\frac{a'(1+a')}{4}\left[j\left(\arccos k, 0\right) + j\left(-\arccos k, 0\right)\right] = \begin{pmatrix} 0 & 0 \\ \frac{\beta}{\alpha} & 0 \end{pmatrix} \tag{13}$$

$$\frac{1+a'}{4\sqrt{a'}}\left[j\left(\arccos k, 0\right) - j\left(-\arccos k, 0\right)\right] = \begin{pmatrix} 0 & \frac{\alpha}{\beta} i \\ \frac{\beta}{\alpha}\frac{2+a'}{a'} i & 0 \end{pmatrix}. \tag{14}$$

These are well-defined as we have $a' > 0$ in this case. Clearly matrices (11) and (13) span the space of all matrices with a single complex entry in the bottom left hand corner. Hence, when combined with matrices (12) and (14), they clearly span the space of all matrices with complex entries in the off diagonal elements.

Subcase B: $a' < 0$ and $a' \neq -1$, i.e., $-1 < 1/k < 1$. This subcase follows similarly; consider the matrices

$$\frac{a'(1-a')}{4}\left[j\left(\frac{\pi}{2}, \arcsin\frac{1}{k}\right) + j\left(\frac{\pi}{2}, \pi - \arcsin\frac{1}{k}\right)\right] = \begin{pmatrix} 0 & 0 \\ \frac{\beta}{\alpha} i & 0 \end{pmatrix} \tag{15}$$

$$\frac{1+a'}{4\sqrt{-a'}}\left[j\left(\frac{\pi}{2}, \arcsin\frac{1}{k}\right) - j\left(\frac{\pi}{2}, \pi - \arcsin\frac{1}{k}\right)\right] = \begin{pmatrix} 0 & \frac{\alpha}{\beta} \\ -\frac{\beta}{\alpha}\frac{1+a'}{1-a'} & 0 \end{pmatrix} \tag{16}$$

and

$$\frac{-a'(1-a')}{4}\left[j\left(0, \arccos\frac{1}{k}\right) + j\left(0, -\arccos\frac{1}{k}\right)\right] = \begin{pmatrix} 0 & 0 \\ \frac{\beta}{\alpha} & 0 \end{pmatrix} \tag{17}$$

$$\frac{1+a'}{4\sqrt{-a'}}\left[j\left(0, \arccos\frac{1}{k}\right) - j\left(0, -\arccos\frac{1}{k}\right)\right] = \begin{pmatrix} 0 & \frac{\alpha}{\beta} i \\ \frac{\beta}{\alpha}\frac{1+a'}{1-a'} i & 0 \end{pmatrix}. \tag{18}$$

These are well-defined as we have $a' < 0$ in this case, as well as $a' \neq -1$. Again, clearly we have that (15) and (17) span all matrices with a single complex entry in the bottom left of the matrix. Hence, adding in (16) and (18), we span all off-diagonal complex matrices, which is what we wanted to show.

In either subcase, our j matrices span all matrices of the form

$$\begin{pmatrix} 0 & A + Bi \\ C + Di & 0 \end{pmatrix}$$

where $A, B, C, D \in \mathbb{R}$. Additionally, our g and h matrices are also in $\mathfrak{g}$, and clearly combining these with the j matrices increases the span to

$$\begin{pmatrix} Ei & A+Bi \\ C+Di & -Ei \end{pmatrix}$$

where $A, B, C, D, E \in \mathbb{R}$. This is a five-dimensional subspace of $\mathfrak{sl}(2, \mathbb{C})$. Now to show that we can span all 6 dimensions of $\mathfrak{sl}(2, \mathbb{C})$, we invoke the fact that $\mathfrak{g}$ is closed under commutation, so $\mathfrak{g}$ contains $[\left(\begin{smallmatrix} 0 & 1 \\ 0 & 0 \end{smallmatrix}\right), \left(\begin{smallmatrix} 0 & 0 \\ 1 & 0 \end{smallmatrix}\right)] = \left(\begin{smallmatrix} 1 & 0 \\ 0 & -1 \end{smallmatrix}\right)$. Hence $\mathfrak{g}$ must include all matrices of the form

$$\begin{pmatrix} F+Ei & A+Bi \\ C+Di & -F-Ei \end{pmatrix}$$

where $A, B, C, D, E, F \in \mathbb{R}$. In other words, $\mathfrak{g} = \mathfrak{sl}(2, \mathbb{C})$.

We've now shown Claim C.5 in the case where $a' \neq 0$ and $a' \neq -1$. We now prove the claim in these remaining two cases.

Case 2: $a' = 0$. In this case we have

$$g(\theta') = -\frac{1}{2}\begin{pmatrix} 0 & 0 \\ \frac{\beta}{\alpha}2e^{-i\theta'} & 0 \end{pmatrix}$$

As θ varies these matrices clearly span all matrices a single complex number in the bottom left entry. Now in this case we also have that

$$h(\theta') = -\frac{1}{2}\begin{pmatrix} -i & \frac{\alpha}{\beta}e^{i\theta'} \\ \frac{\beta}{\alpha}e^{-i\theta'} & i \end{pmatrix}$$

Since $\mathfrak{g}$ is closed under addition and scalar multiplication by $\mathbb{R}$, and applying

$$h(\theta') - h(\theta'') = -\frac{1}{2}\begin{pmatrix} 0 & \frac{\alpha}{\beta}(e^{i\theta'} - e^{i\theta''}) \\ \frac{\beta}{\alpha}(e^{-i\theta'} - e^{i\theta''}) & 0 \end{pmatrix} \in \mathfrak{g}$$

Now adding in multiples of g, we have that $\mathfrak{g}$ contains matrices of the form

$$\begin{pmatrix} 0 & \frac{\alpha}{\beta}(e^{i\theta'} - e^{i\theta''}) \\ 0 & 0 \end{pmatrix}$$

which clearly span all matrices with a complex entry in the upper right corner. Hence we span all off-diagonal matrices. Now adding in $h(\theta)$ for any θ, we span all matrices of the form $\begin{pmatrix} Ei & A+Bi \\ C+Di & -Ei \end{pmatrix}$ where $A, B, C, D, E \in \mathbb{R}$. As discussed in Case 1, by taking the closure of these under commutation we have that $\mathfrak{g} = \mathfrak{sl}(2\mathbb{C})$ as desired, which completes the proof of Case 2.

Case 3: $a' = -1$. This case follows very similarly to Case 2. When $a' = -1$ we have that

$$h(\theta') = -\frac{1}{2}\begin{pmatrix} 0 & \frac{\alpha}{\beta}2e^{i\theta'} \\ 0 & 0 \end{pmatrix}$$

which clearly span all complex matrices with a single entry in the upper right corner. In this case, we also have that

$$g(\theta') = -\frac{1}{2}\begin{pmatrix} i & -\frac{\alpha}{\beta} - e^{i\theta'} \\ \frac{\beta}{\alpha}e^{-i\theta'} & -i \end{pmatrix},$$

By considering the difference $g(\theta') - g(\theta'')$, and noting that we already span matrices with a single entry in the upper right corner, this shows that we span all off-diagonal matrices. Now adding in $g(\theta')$ for any θ' we see that we span all matrices of the form $\begin{pmatrix} Ei & A+Bi \\ C+Di & -Ei \end{pmatrix}$ where $A, B, C, D, E \in \mathbb{R}$. As discussed in Case 1, by taking the closure of these under commutation we have that $\mathfrak{g} = \mathfrak{sl}(2, \mathbb{C})$ as desired. This completes the proof of Case 3, hence the proof of the claim. ◀

Identity Testing for Constant-Width, and Commutative, Read-Once Oblivious ABPs

Rohit Gurjar[*1], **Arpita Korwar**[2], **and Nitin Saxena**[†3]

1 **Aalen University, Aalen, Germany**
rgurjar@cse.iitk.ac.in
2 **Department of Computer Science and Engineering, IIT Kanpur, Kanpur, India**
arpk@cse.iitk.ac.in
3 **Department of Computer Science and Engineering, IIT Kanpur, Kanpur, India**
nitin@cse.iitk.ac.in

Abstract

We give improved hitting-sets for two special cases of Read-once Oblivious Arithmetic Branching Programs (ROABP). First is the case of an ROABP with known variable order. The best hitting-set known for this case had cost $(nw)^{O(\log n)}$ where n is the number of variables and w is the width of the ROABP. Even for a constant-width ROABP, nothing better than a quasi-polynomial bound was known. We improve the hitting-set complexity for the known-order case to $n^{O(\log w)}$. In particular, this gives the first polynomial time hitting-set for constant-width ROABP (known-order). However, our hitting-set works only over those fields whose characteristic is zero or large enough. To construct the hitting-set, we use the concept of the rank of partial derivative matrix. Unlike previous approaches whose starting point is a monomial map, we use a polynomial map directly.

The second case we consider is that of commutative ROABP. The best known hitting-set for this case had cost $d^{O(\log w)}(nw)^{O(\log\log w)}$, where d is the individual degree. We improve this hitting-set complexity to $(ndw)^{O(\log\log w)}$. We get this by achieving rank concentration more efficiently.

1998 ACM Subject Classification F.1.3 Complexity Measures and Classes, F.2.1 Numerical Algorithms and Problems

Keywords and phrases PIT, hitting-set, constant-width ROABPs, commutative ROABPs

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.29

1 Introduction

The polynomial identity testing (PIT) problem asks if a given multivariate polynomial is identically zero. The input to the problem is given via an arithmetic model computing a polynomial, for example, an arithmetic circuit or an arithmetic branching program. These are arithmetic analogues of boolean circuits and boolean branching programs, respectively. The degree of the given polynomial is assumed to be polynomially bounded in the circuit size. Usually, any such circuit or branching program can compute a polynomial with exponentially many monomials (exponential in the circuit size). Thus, one cannot compute the polynomial explicitly in an efficient way. However, given such an input, it is possible to efficiently evaluate the polynomial at a point in the field. This property enables a randomized polynomial identity

* Supported by DFG grant TH 472/4 and TCS research fellowship.
† Supported by DST-SERB.

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 29; pp. 29:1–29:16

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

test with one-sided error. It is known that evaluating a small-degree nonzero polynomial over a random point gives a nonzero value with a good probability [8, 22, 23]. Thus, the randomized test is to just evaluate the input polynomial, given as an arithmetic circuit or an arithmetic branching program at a random point.

Finding an efficient deterministic algorithm for PIT has been a major open question in complexity theory. The question is also related to arithmetic circuit lower bounds [1, 12, 15]. The PIT problem has been studied in two paradigms: (i) blackbox test, where one can only evaluate the polynomial at a chosen point, (ii) whitebox test, where one has access to the input circuit or arithmetic branching program. A blackbox test is essentially the same as finding a hitting-set – a set of points such that any nonzero polynomial evaluates to a nonzero value on at least one of the points in the set. This work concerns finding hitting-sets for a special model, called read-once oblivious arithmetic branching programs (ROABP).

An *arithmetic branching program (ABP)* is a directed layered graph, with edges going from a layer of vertices to the next layer. The first and the last layers have one vertex each, called the source and the sink. Each edge of the graph has a label, which is a simple polynomial, for example a univariate polynomial. For any path p, its weight is defined to be the product of labels on all the edges in p. The ABP is said to compute a polynomial which is the sum of weights of all the paths from the source to the sink. ABPs are a strong model for computing polynomials. It is known that for any arithmetic circuit with polynomially bounded degree, one can find an ABP of quasi-polynomial size computing the same polynomial (see for example [17]). Apart from its size, another important parameter for an ABP is its width. The width of an ABP is the maximum number of vertices in any layer of the associated graph. Even when the the width is restricted to a constant, the ABP model is quite powerful. Ben-Or and Cleve [6] have shown that width-3 ABPs have the same expressive power as arithmetic formulas.

An ABP is called a *read-once oblivious ABP or ROABP* if every variable occurs in at most one layer of edges in the ABP. For an ROABP, one can assume without loss of generality that any variable occurs in exactly one layer of edges. The order of the variables in consecutive layers is said to be the *variable order* of the ROABP. The read-once property severely restricts the power of the ABP. There are polynomials known which can be computed by a simple depth-3 ($\Sigma\Pi\Sigma$) circuit but require an exponential size ROABP [16]. Also note that there are polynomials which have a small ROABP in one variable order but require exponential size in another variable order. Nisan [19] gave the exact characterization of the polynomials computed by width-w ROABPs in a certain variable order. In particular, they gave exponential lower bounds for this model. Their work is actually on non-commutative ABPs but the same results also apply to ROABP.

The question of whitebox identity testing of ROABPs has been settled by Raz and Shpilka [21], who gave a polynomial time algorithm for this. However, though ROABPs are a relatively well-understood model, we still do not have a polynomial time blackbox algorithm. The blackbox question is studied with two variations: one where we know the variable order of the ROABP and the other where we do not know it. For known-order ROABPs, Forbes and Shpilka [10] gave the first efficient blackbox test with $(ndw)^{O(\log n)}$ time complexity, where n is the number of variables, w is the width of the ROABP, and, d is the individual degree bound of each variable. For the unknown-order case, Forbes et al. [9] gave an $n^{O(d \log w \log n)}$-time blackbox test. Observe that their complexity is quasi-polynomial only when d is small. Subsequently, Agrawal et al. [2] removed the exponential dependence on the individual degree. They gave an $(ndw)^{O(\log n)}$-time blackbox test for the unknown-order case. Note that these results remain quasi-polynomial even in the case of constant width.

Studying ROABPs has also led to PIT results for other computational models, for example, sub-exponential size hitting-sets for depth-3 multilinear circuits [7] and sub-exponential time whitebox test for read-k oblivious ABPs [4]. It is possible that the results and techniques for ROABPs can help solve the PIT problem for more general models.

Another motivation to study ROABPs comes from their boolean analogues, called read-once ordered branching programs (ROBP). ROBPs have been studied extensively, with regard to the RL versus L question (randomized log-space versus log-space). The problem of finding hitting-sets for ROABP can be viewed as an analogue of finding pseudorandom generators (PRG) for ROBP. A pseudorandom generator for a boolean function f is an algorithm which can generate a probability distribution (with a small sample space) with the property that f cannot distinguish it from the uniform random distribution (see [5] for details). Constructing an optimal PRG for ROBP, i.e., with $O(\log n)$ seed length or polynomial size sample space, would imply RL = L. This question has similar results as those for PIT of ROABPs, though no connection is known between the two questions. The best known PRG is of seed length $O(\log^2 n)$ ($n^{O(\log n)}$ size sample space), when variable order is known [18, 14, 20]. On the other hand, in the unknown-order case, the best known seed length is of size $n^{1/2+o(1)}$ [13]. Finding an $O(\log n)$-seed PRG even for constant-width known-order ROBPs has been a challenging open question.

Our first result addresses the analogous question in the arithmetic setting. We give the first polynomial time blackbox test for constant-width known-order ROABPs. However, it works only for zero or large characteristic fields. Our idea is inspired from the pseudorandom construction of Impagliazzo, Nisan and Wigderson [14] for ROBPs. While their result does not give better PRGs for the constant-width case, we are able to achieve this in the arithmetic setting.

▶ **Theorem** (Theorem 3.6). *Let $\mathcal{C}$ be the class of n-variate, individual degree d polynomials in $\mathbb{F}[\boldsymbol{x}]$ computed by a width-w ROABP in the variable order $(x_1, x_2, \ldots, x_n)$. Then there is a $dn^{O(\log w)}$-time hitting-set for $\mathcal{C}$, when $\mathrm{char}(\mathbb{F}) = 0$ or $\mathrm{char}(\mathbb{F}) > ndw^{\log n}$.*

Our test actually works for any width. Its time complexity is better than the previous results on ROABP, when $w < n$ and is same in the other case. Our main technique uses the notion of rank of the partial derivative matrix defined by Nisan [19]. We show that for a nonzero bivariate polynomial $f(x_1, x_2)$ computed by a width-w ROABP, the univariate polynomial $f(t^w, t^w + t^{w-1})$ is nonzero. Our argument is that any bivariate polynomial which becomes zero on $(t^w, t^w + t^{w-1})$ has rank more than w, while a polynomial computed by a width-w ROABP has rank w or less. Then, we use the map $(x_1, x_2) \mapsto (t^w, t^w + t^{w-1})$ recursively in $\log n$ rounds to achieve the above mentioned hitting-set. Our technique has a crucial difference from the previous works on ROABPs [9, 10, 2]. The starting point in all the previous techniques is a monomial map, i.e., each variable is mapped to a univariate monomial. On the other hand, we argue with a polynomial map directly (where each variable is mapped to a univariate polynomial). Our approach can potentially lead to a polynomial time hitting-set for ROABPs. The goal would be to obtain a univariate n-tuple $(p_1(t), \ldots, p_n(t))$, such that any polynomial which becomes zero on $(p_1(t), \ldots, p_n(t))$ must have rank or evaluation dimension higher than w. We conjecture that $(t^r, (t+1)^r, \ldots, (t+n-1)^r)$ is one such tuple, where r is polynomially large (Conjecture 3.8).

It is also possible that our ideas for the arithmetic setting can help constructing an optimal PRG for constant-width ROBP.

Our second result is for a special case of ROABPs, called commutative ROABPs. An ROABP is commutative if its edge layers can be exchanged without affecting the polynomial computed. In particular, if all paths from the source to the sink are vertex disjoint, then the

ROABP is commutative. Note that for a commutative ROABP, knowing the variable order is irrelevant. Commutative ROABPs have slightly better hitting-sets than the general case, but still no polynomial time hitting-set is known. The previously best known hitting-set for them has time complexity $d^{O(\log w)}(nw)^{O(\log\log w)}$ [9]. We improve this to $(ndw)^{O(\log\log w)}$.

▶ **Theorem** (Theorem 4.10). *There is an $(ndw)^{O(\log\log w)}$-time hitting-set for n-variate commutative ROABPs with width w and individual degree d.*

To get this result we follow the approach of Forbes et al. [9], which uses the notion of rank concentration. We achieve rank concentration more efficiently using the basis isolation technique of Agrawal et al. [2]. The same technique also yields a more efficient concentration in depth-3 set-multilinear circuits (see Section 2 for the definition). However, it is not clear if it gives better hitting-sets for them. The best known hitting-set for them has complexity $n^{O(\log n)}$ [3].

2 Preliminaries

2.1 Definitions and Notations

$\mathbb{N}$ denotes the set of all non-negative integers, i.e., $\{0, 1, 2, \dots\}$. $[n]$ denotes the set $\{1, 2, \dots, n\}$. $[\![d]\!]$ denotes the set $\{0, 1, \dots, d\}$. $\boldsymbol{x}$ will denote a set of variables, usually the set $\{x_1, x_2, \dots, x_n\}$. For a set of n variables $\boldsymbol{x}$ and for an exponent $\boldsymbol{a} = (a_1, a_2, \dots, a_n) \in \mathbb{N}^n$, $\boldsymbol{x}^{\boldsymbol{a}}$ will denote the monomial $\prod_{i=1}^{n} x_i^{a_i}$. The *support* of a monomial $\boldsymbol{x}^{\boldsymbol{a}}$, denoted by $\mathrm{Supp}(\boldsymbol{a})$, is the set of variables appearing in that monomial, i.e., $\{x_i \mid i \in [n], a_i > 0\}$. The *support size* of a monomial is the cardinality of its support, denoted by $\mathrm{supp}(\boldsymbol{a})$. A monomial is said to be ℓ-support if its support size is ℓ. For a polynomial $P(\boldsymbol{x})$, the coefficient of a monomial $\boldsymbol{x}^{\boldsymbol{a}}$ in $P(\boldsymbol{x})$ is denoted by $\mathrm{coef}_P(\boldsymbol{x}^{\boldsymbol{a}})$. In particular, $\mathrm{coef}_P(1)$ denotes the constant term of the polynomial P.

For a monomial $\boldsymbol{x}^{\boldsymbol{a}}$, $\sum_i a_i$ is said to be its *degree* and a_i is said to be its *degree in variable* x_i for each i. Similarly for a polynomial P, its degree (or degree in x_i) is the maximum degree (or maximum degree in x_i) of any monomial in P with a nonzero coefficient. We define the *individual degree* of P to be $\text{indv-deg}(P) = \max_i\{\deg_{x_i}(P)\}$, where $\deg_{x_i}$ denotes degree in x_i.

To better understand polynomials computed by ROABPs, we often use polynomials over an algebra $\mathbb{A}$, i.e., polynomials whose coefficients come from $\mathbb{A}$. Matrix algebra is the vector space of matrices equipped with the matrix product. $\mathbb{F}^{m\times n}$ represents the set of all $m \times n$ matrices over the field $\mathbb{F}$. Note that the algebra of $w \times w$ matrices, has dimension w^2.

We often view a vector/matrix with polynomial entries, as a polynomial with vector/matrix coefficients. For example,

$$D(x,y) = \begin{pmatrix} 1+x & y-xy \\ x+y & 1+xy \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} 1 + \begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix} x + \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} y + \begin{pmatrix} 0 & -1 \\ 0 & 1 \end{pmatrix} xy.$$

Here, the coef_D operator will return a matrix for any monomial, for example, $\mathrm{coef}_D(y) = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$. For a polynomial $D(\boldsymbol{x}) \in \mathbb{A}[\boldsymbol{x}]$ over an algebra, its *coefficient space* is the space spanned by its coefficients.

For a matrix R, $R(i, j)$ denotes its entry in the i-th row and j-th column.

As mentioned earlier, a deterministic blackbox PIT is equivalent to constructing a hitting-set. A set of points $\mathcal{H} \in \mathbb{F}^n$ is called a *hitting-set* for a class $\mathcal{C}$ of n-variate polynomials if for any nonzero polynomial P in $\mathcal{C}$, there exists a point in $\mathcal{H}$ where P evaluates to a nonzero

value. An $f(n)$-time hitting-set would mean that the hitting-set can be generated in time $f(n)$ for input size n.

2.2 Arithmetic Branching Programs

An ABP is a directed graph with $q+1$ layers of vertices $\{V_0, V_1, \ldots, V_q\}$ and a start node u and an end node t such that the edges are only going from u to V_0, V_{i-1} to V_i for any $i \in [q]$ and V_q to t. The edges have univariate polynomials as their weights and as a convention, the edges going from u and those coming to t have weights from the field $\mathbb{F}$. The ABP is said to compute the polynomial $C(\boldsymbol{x}) = \sum_{p \in \text{paths}(u,t)} \prod_{e \in p} W(e)$, where $W(e)$ is the weight of the edge e.

The ABP has width w if $|V_i| \leq w$ for all $i \in [\![q]\!]$. Without loss of generality we can assume $|V_i| = w$ for each $i \in [\![q]\!]$.

It is well-known that the sum over all paths in a layered graph can be represented by an iterated matrix multiplication. To see this, let the set of nodes in V_i be $\{v_{i,j} \mid j \in [w]\}$. It is easy to see that the polynomial computed by the ABP is the same as $U^{\mathsf{T}}(\prod_{i=1}^{q} D_i)T$, where $U, T \in \mathbb{F}^{w \times 1}$ and D_i is a $w \times w$ matrix for $1 \leq i \leq q$ such that

$$\begin{aligned} U(\ell) &= W(u, v_{0,\ell}) \text{ for } 1 \leq \ell \leq w \\ D_i(k,\ell) &= W(v_{i-1,k}, v_{i,\ell}) \text{ for } 1 \leq \ell, k \leq w \text{ and } 1 \leq i \leq q \\ T(k) &= W(v_{q,k}, t) \text{ for } 1 \leq k \leq w \end{aligned}$$

2.2.1 Read-once Oblivious ABP

An ABP is called a *read-once oblivious ABP (ROABP)* if the edge weights in different layers are univariate polynomials in distinct variables. Formally, the entries in D_i come from $\mathbb{F}[x_{\pi(i)}]$ for all $i \in [q]$, where π is a permutation on the set $[q]$. Here, q is the same as n, the number of variables. The order $(x_{\pi(1)}, x_{\pi(2)}, \ldots, x_{\pi(n)})$ is said to be the variable order of the ROABP.

Viewing $D_i(x_{\pi(i)}) \in \mathbb{F}^{w \times w}[x_{\pi(i)}]$ as a polynomial over the matrix algebra, we can write the polynomial computed by an ROABP as

$$C(\boldsymbol{x}) = U^{\mathsf{T}} D_1(x_{\pi(1)}) D_2(x_{\pi(2)}) \cdots D_n(x_{\pi(n)}) T\,.$$

An equivalent representation of a width-w ROABP can be

$$C(\boldsymbol{x}) = D_1(x_{\pi(1)}) D_2(x_{\pi(2)}) \cdots D_n(x_{\pi(n)})\,,$$

where $D_1 \in \mathbb{F}^{1 \times w}[x_{\pi(1)}]$, $D_i \in \mathbb{F}^{w \times w}[x_{\pi(i)}]$ for $2 \leq i \leq n-1$ and $D_n \in \mathbb{F}^{w \times 1}[x_{\pi(n)}]$.

2.2.2 Commutative ROABP

An ROABP $U^{\mathsf{T}} \left(\prod_{i=1}^{q} D_i\right) T$ is a commutative ROABP, if all D_is are polynomials over a commutative subalgebra of the matrix algebra. For example, if the coefficients in the polynomials D_is are all diagonal matrices. Note that the order of the variables becomes insignificant for a commutative ROABP. A polynomial computed by a commutative ROABP can be computed by an ROABP in any variable order.

2.2.3 Set-multilinear Circuits

A depth-3 set-multilinear circuit is a circuit of the form

$$C(\boldsymbol{x}) = \sum_{i=1}^{k} l_{i,1}(\boldsymbol{x}_1)\, l_{i,2}(\boldsymbol{x}_2) \cdots\ l_{i,q}(\boldsymbol{x}_q)\,,$$

where $l_{i,j}$s are linear polynomials and $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_q$ form of partition of $\boldsymbol{x}$. It is known that these circuits are subsumed by ROABPs [9]. However, they are incomparable to commutative ROABPs. Consider the corresponding polynomial over a k-dimensional algebra

$$D(\boldsymbol{x}) = D_1(\boldsymbol{x}_1)D_2(\boldsymbol{x}_2)\cdots D_q(\boldsymbol{x}_q),$$

where $D_j = (l_{1,j}, l_{2,j}, \ldots, l_{k,j})$ and the algebra product is coordinate-wise product. It is easy to see that $C = (1, 1, \ldots, 1) \cdot D$. Note that the polynomials D_is are over a commutative algebra. Hence, some of our techniques for commutative ROABPs also work for set-multilinear circuits.

3 Hitting-set for Known-order ROABP

3.1 Bivariate ROABP

To construct a hitting-set for ROABPs, we start with the bivariate case. Recall that a bivariate ROABP is of the form $U^{\mathsf{T}} D_1(x_1) D_2(x_2) T$, where $U, T \in \mathbb{F}^{w\times 1}$, $D_1 \in \mathbb{F}^{w\times w}[x_1]$ and $D_2 \in \mathbb{F}^{w\times w}[x_2]$. It is easy to see that a bivariate polynomial $f(x_1, x_2)$ computed by a width-w ROABP can be written as $f(x_1, x_2) = \sum_{r=1}^{w} g_r(x_1)h_r(x_2)$. To give a hitting-set for this, we will use the notion of a partial derivative matrix defined by Nisan [19] in the context of lower bounds. Let $f \in \mathbb{F}[x_1, x_2]$ have its individual degree bounded by d. The *partial derivative matrix* M_f for f is a $(d+1)\times(d+1)$ matrix with

$$M_f(i, j) = \mathrm{coef}_f(x_1^i x_2^j) \in \mathbb{F},$$

for all $i, j \in [\![d]\!]$. It is known that the rank of M_f is equal to the smallest possible width of an ROABP computing f [19].

▶ **Lemma 3.1** (rank ≤ width). *For any polynomial $f(x_1, x_2) = \sum_{r=1}^{w} g_r(x_1)h_r(x_2)$, $\mathrm{rank}(M_f) \leq w$.*

Proof. Let us define $f_r = g_r h_r$, for all $r \in [w]$. Clearly, $M_f = \sum_{r=1}^{w} M_{f_r}$, as $f = \sum_{r=1}^{w} f_r$. We will show that $\mathrm{rank}(M_{f_r}) \leq 1$, for all $r \in [w]$. As $f_r = g_r(x_1)h_r(x_2)$, its coefficients can be written as a product of coefficients from g_r and h_r, i.e.,

$$\mathrm{coef}_{f_r}(x_1^i x_2^j) = \mathrm{coef}_{g_r}(x_1^i)\,\mathrm{coef}_{h_r}(x_2^j)\,.$$

Now, it is easy to see that

$$M_{f_r} = u_r v_r^{\mathsf{T}}\,,$$

where $u_r, v_r \in \mathbb{F}^{d+1}$ with $u_r = (\mathrm{coef}_{g_r}(x_1^i))_{i=0}^{d}$ and $v_r = (\mathrm{coef}_{h_r}(x_2^i))_{i=0}^{d}$.

Thus, $\mathrm{rank}(M_{f_r}) \leq 1$ and $\mathrm{rank}(M_f) \leq w$. ◀

One can also show that if $\mathrm{rank}(M_f) = w$ then there exists a width-w ROABP computing f. We skip this proof as we will not need it. Now, using the above lemma we give a hitting-set for bivariate ROABPs.

▶ **Lemma 3.2.** *Let* $\mathrm{char}(\mathbb{F}) = 0$*, or* $\mathrm{char}(\mathbb{F}) > d$*. Let* $f(x_1, x_2) = \sum_{r=1}^{w} g_r(x_1)h_r(x_2)$ *be a nonzero bivariate polynomial over* $\mathbb{F}$ *with individual degree* d*. Then* $f(t^w, t^w + t^{w-1}) \neq 0$*.*

Proof. Let $f'(t)$ be the polynomial after the substitution, i.e., $f' = f(t^w, t^w + t^{w-1})$. Any monomial $x_1^i x_2^j$ will be mapped to the polynomial $t^{wi}(t^w + t^{w-1})^j$, under the mentioned substitution. The highest power of t coming from this polynomial is $t^{w(i+j)}$. We will cluster together all the monomials for which this highest power is the same, i.e., $i + j$ is the same. The coefficients corresponding to any such cluster of monomials will form a *diagonal* in M_f. The set $\{M_f(i, j) \mid i + j = k\}$ is defined to be the *k-th diagonal* of M_f, for all $0 \leq k \leq 2d$. Let ℓ be the highest number such that ℓ-th diagonal has at least one nonzero element, i.e.,

$$\ell = \max\{i + j \mid M_f(i, j) \neq 0\}\,.$$

As $\mathrm{rank}(M_f) \leq w$ (from Lemma 3.1), we claim that the ℓ-th diagonal has at most w nonzero elements. To see this, let $\{(i_1, j_1), (i_2, j_2), \ldots, (i_{w'}, j_{w'})\}$ be the set of indices where the ℓ-th diagonal of M_f has nonzero elements, i.e., the set $\{(i, j) \mid M_f(i, j) \neq 0,\ i + j = \ell\}$. As $M_f(i, j) = 0$ for any $i + j > \ell$, it is easy to see that the rows $\{M_f(i_1), M_f(i_2), \ldots, M_f(i_{w'})\}$ are linearly independent. Thus, $w' \leq \mathrm{rank}(M_f) \leq w$.

Now, we claim that there exists an r with $w(\ell - 1) < r \leq w\ell$ such that $\mathrm{coef}_{f'}(t^r) \neq 0$. To see this, first observe that the highest power of t which any monomial $x_1^i x_2^j$ with $i + j < \ell$ can contribute is $t^{w(\ell-1)}$. Thus, for any $w(\ell - 1) < r \leq w\ell$, the term t^r can come only from the monomials $x_1^i x_2^j$ with $i + j \geq \ell$. We can ignore the monomials $x_1^i x_2^j$ with $i + j > \ell$ as $\mathrm{coef}_f(x_1^i x_2^j) = M_f(i, j) = 0$, when $i + j > \ell$. Now, for any $i + j = \ell$, the monomial $x_1^i x_2^j$ goes to

$$t^{w(\ell-j)}(t^w + t^{w-1})^j = \sum_{p=0}^{j} \binom{j}{p} t^{w\ell-p}\,.$$

Hence, for any $0 \leq p < w$,

$$\mathrm{coef}_{f'}(t^{w\ell-p}) = \sum_{a=1}^{w'} M_f(i_a, j_a)\binom{j_a}{p}\,.$$

Writing this in the matrix form we get

$$[\mathrm{coef}_{f'}(t^{w\ell}) \ \cdots\ \mathrm{coef}_{f'}(t^{w\ell-w+1})] = [M_f(i_1, j_1) \ \cdots\ M_f(i_{w'}, j_{w'})]C\,,$$

where C is a $w' \times w$ matrix with $C(a, b) = \binom{j_a}{b-1}$, for all $a \in [w']$ and $b \in [w]$. If all the rows of C are linearly independent then clearly, $\mathrm{coef}_{f'}(t^r) \neq 0$ for some $w(\ell - 1) < r \leq w\ell$. We show the linear independence in Claim 3.3. To show this linear independence we need to assume that the numbers $\{j_a\}_a$ are all distinct. Hence, we need the field characteristic to be zero or strictly greater than d, as j_a can be as high as d for some $a \in [w']$.

▶ **Claim 3.3.** *Let* C *be a* $w \times w$ *matrix with* $C(a, b) = \binom{j_a}{b-1}$*, for all* $a \in [w]$ *and* $b \in [w]$*, where* $\{j_a\}_a$ *are all distinct numbers. Then* C *has full rank.*

Proof. We will show that for any nonzero vector $\alpha := (\alpha_1, \alpha_2 \ldots, \alpha_w) \in \mathbb{F}^{w\times 1}$, $C\alpha \neq 0$. Consider the polynomial $h(y) = \sum_{b=1}^{w} \alpha_b \frac{y(y-1)\cdots(y-b+2)}{(b-1)!}$. As $h(y)$ is a nonzero polynomial with degree bounded by $w - 1$, it can have at most $w - 1$ roots. Thus, there exists an $a \in [w]$ such that $h(j_a) = \sum_{b=1}^{w} \alpha_b \binom{j_a}{b-1} \neq 0$. ◀

◀

As mentioned above, the hitting-set proof works only when the field characteristic is zero or greater than d. We given an example over a small characteristic field, which demonstrates that the problem is not with the proof technique, but with the hitting-set itself. Let the field characteristic be 2. Consider the polynomial $f(x_1, x_2) = x_2^2 + x_1^2 + x_1$. Clearly, f has a width-2 ROABP. For a width-2 ROABP, the map in Lemma 3.2 would be $(x_1, x_2) \mapsto (t^2, t^2 + t)$. However, $f(t^2, t^2 + t) = 0$ (over $\mathbb{F}_2$). Hence, the hitting-set does not work.

Now, we move on to getting a hitting-set for an n-variate ROABP.

3.2 n-variate ROABP

Observe that the map given in Lemma 3.2 works irrespective of the degree of the polynomial, as long as the field characteristic is large enough. We plan to obtain a hitting-set for general n-variate ROABP by applying this map recursively. For this, we use the standard divide and conquer technique. First, we make pairs of consecutive variables in the ROABP. For each pair (x_{2i-1}, x_{2i}), we apply the map from Lemma 3.2, using a new variable t_i. Thus, we go to $n/2$ variables from n variables. In Lemma 3.4, we use a hybrid argument to show that after this substitution the polynomial remains nonzero. Moreover, the new polynomial can be computed by a width-w ROABP. Thus, we can again use the same map on pairs of new variables. By repeating the halving procedure $\log n$ times we get a univariate polynomial. In each round the degree of the polynomial gets multiplied by w. Hence, after $\log n$ rounds, the degree of the univariate polynomial is bounded by $w^{\log n}$ times the original degree. Without loss of generality, let us assume that n is a power of 2.

▶ **Lemma 3.4** (Halving the number of variables). *Let* $\mathrm{char}(\mathbb{F}) = 0$, *or* $\mathrm{char}(\mathbb{F}) > d$. *Let* $f(\boldsymbol{x}) = D_1(x_1)D_2(x_2)\cdots D_n(x_n)$ *be a nonzero polynomial computed by a width-*w *and individual degree-*d *ROABP, 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 all* $2 \le i \le n-1$. *Let the map* $\phi\colon \boldsymbol{x} \to \mathbb{F}[\boldsymbol{t}]$ *be such that for any index* $1 \le i \le n/2$,

$$\begin{aligned} \phi(x_{2i-1}) &= t_i^w, \\ \phi(x_{2i}) &= t_i^w + t_i^{w-1}. \end{aligned}$$

Then $f(\phi(\boldsymbol{x})) \neq 0$. *Moreover, the polynomial* $f(\phi(\boldsymbol{x})) \in \mathbb{F}[t_1, t_2, \ldots, t_{n/2}]$ *is computed by a width-*w *ROABP in the variable order* $(t_1, t_2, \ldots, t_{n/2})$.

Proof. Let us apply the map in $n/2$ rounds, i.e., define a sequence of polynomials $(f = f_0, f_1, \ldots, f_{n/2} = f(\phi(\boldsymbol{x})))$ such that the polynomial f_i is obtained by making the replacement $(x_{2i-1}, x_{2i}) \mapsto (\phi(x_{2i-1}), \phi(x_{2i}))$ in f_{i-1} for each $1 \le i \le n/2$. We will show that for each $1 \le i \le n/2$, if $f_{i-1} \neq 0$ then $f_i \neq 0$. Clearly this proves the first part of the lemma.

Note that f_{i-1} is a polynomial over variables $\{t_1, \ldots, t_{i-1}, x_{2i-1}, \ldots, x_n\}$. As $f_{i-1} \neq 0$, there exists a constant tuple $\alpha \in \mathbb{F}^{n-i-1}$ such that after replacing the variables $(t_1, \ldots, t_{i-1}, x_{2i+1}, \ldots, x_n)$ with α, f_{i-1} remains nonzero. After this replacement we get a polynomial f'_{i-1} in the variables (x_{2i-1}, x_{2i}). As f is computed by the ROABP $D_1D_2\cdots D_n$, the polynomial f'_{i-1} can be written as $U^{\mathsf{T}}D_{2i-1}(x_{2i-1})D_{2i}(x_{2i})T$ for some $U, T \in \mathbb{F}^{w\times 1}$. In other words, f'_{i-1} has a bivariate ROABP of width w. Thus, $f'_{i-1}(\phi(x_{2i-1}), \phi(x_{2i}))$ is nonzero from Lemma 3.2. But, $f'_{i-1}(\phi(x_{2i-1}), \phi(x_{2i}))$ is nothing but the polynomial obtained after replacing the variables $(t_1, \ldots, t_{i-1}, x_{2i+1}, \ldots, x_n)$ in f_i with α. Thus, f_i is nonzero. This finishes the proof.

Now, we argue that $f(\phi(\boldsymbol{x}))$ has a width w ROABP. Let $D'_i := D_{2i-1}(t_i^w)D_{2i}(t_i^w + t_i^{w-1})$ for all $1 \le i \le n/2$. Clearly, $D'_1D'_2\cdots D'_{n/2}$ is an ROABP computing $f(\phi(\boldsymbol{x}))$ in variable order $(t_1, t_2, \ldots, t_{n/2})$, as $D'_1 \in \mathbb{F}^{1\times w}[t_1]$, $D'_{n/2} \in \mathbb{F}^{w\times 1}[t_{n/2}]$ and $D'_i \in \mathbb{F}^{w\times w}[t_i]$ for all $2 \le i \le n/2 - 1$. ◀

By applying the map ϕ in Lemma 3.4, we reduced an n-variate ROABP to an $(n/2)$-variate ROABP, while preserving the non-zeroness. The resulting ROABP has same width w, but the individual degree goes up to become $2dw$, where d is the original individual degree. As our map ϕ is degree insensitive, we can apply the same map again on the variables $\{t_i\}_{i=1}^{n/2}$. It is easy to see that when the map ϕ is repeatedly applied in this way $\log n$ times, we get a nonzero univariate polynomial of degree $ndw^{\log n}$. Next lemma puts it formally. For ease of notation, we use the variable numbering from 0 to $n-1$. Let $p_0(t) = t^w$ and $p_1(t) = t^w + t^{w-1}$.

▶ **Lemma 3.5.** *Let* $\mathrm{char}(\mathbb{F}) = 0$, *or* $\mathrm{char}(\mathbb{F}) \geq ndw^{\log n}$. *Let* $f \in \mathbb{F}[\boldsymbol{x}]$ *be a nonzero polynomial, with individual degree* d, *computed by a width-*w *ROABP in variable order* $(x_0, x_1, \ldots, x_{n-1})$. *Let the map* $\phi\colon \{x_0, x_1, \ldots, x_{n-1}\} \to \mathbb{F}[t]$ *be such that for any index* $0 \leq i \leq n-1$,

$$\phi(x_i) = p_{i_1}(p_{i_2} \cdots (p_{i_{\log n}}(t)))\,,$$

where $i_{\log n}\ i_{\log n - 1} \cdots i_1$ *is the binary representation of* i. *Then* $f(\phi(\boldsymbol{x}))$ *is a nonzero univariate polynomial with degree* $ndw^{\log n}$.

Note that the map ϕ crucially uses the knowledge of the variable order. In the last round when we are going from two variables to one, the individual degree is $ndw^{\log n - 1}$ and Lemma 3.2 requires $\mathrm{char}(\mathbb{F})$ to be higher than the individual degree. Thus, having $\mathrm{char}(\mathbb{F}) \geq ndw^{\log n}$ suffices. For a univariate polynomial, the standard hitting-set is to plug-in distinct field values as many as one more than the degree. Thus, we get the following theorem.

▶ **Theorem 3.6.** *For an* n*-variate, individual degree* d *and width-*w *ROABP, there is a blackbox PIT with time complexity* $O(ndw^{\log n})$, *when the variable order is known and the field characteristic is zero or at least* $ndw^{\log n}$.

From this, we immediately get the following result for constant-width ROABPs. Note that when w is constant, the lower bound on the characteristic also becomes $\mathsf{poly}(n)$.

▶ **Corollary 3.7.** *There is a polynomial time blackbox PIT for constant width ROABPs, with known variable order and field characteristic being zero (or polynomially large).*

As mentioned earlier, our approach can potentially lead to a polynomial time hitting-set for ROABPs. We make the following conjecture for which we hope to get a proof on the lines of Lemma 3.2.

▶ **Conjecture 3.8.** *Let* $\mathrm{char}(\mathbb{F}) = 0$. *Let* $f(\boldsymbol{x}) \in \mathbb{F}[\boldsymbol{x}]$ *be an* n*-variate, degree-*d *polynomial computed by a width-*w *ROABP. Then* $f(t^r, (t+1)^r, \ldots, (t+n-1)^r) \neq 0$ *for some* r *bounded by* $\mathsf{poly}(n, w, d)$.

4 Commutative ROABP

In this section, we give better hitting-sets for commutative ROABPs. Recall that an ROABP is commutative if the matrices involved in the matrix product come from a commutative algebra. To elaborate, a commutative ROABP is of the form $U^{\mathsf{T}} D_1 D_2 \cdots D_n T$, where $U, T \in \mathbb{F}^{w \times 1}$ and $D_i \in \mathbb{F}^{w \times w}[x_i]$ is a polynomial over a commutative subalgebra of $\mathbb{F}^{w \times w}$ for each i. In simple words, $D_i D_j = D_j D_i$ for any $i, j \in [n]$. As the order of variables does not matter for a commutative ROABP, we take the standard variable order $(x_1, x_2, \ldots, x_n)$. Here we work with the polynomial $D = D_1 D_2 \cdots D_n$ over the matrix algebra. With an abuse of notation, we say $D_1 D_2 \cdots D_n$ is an ROABP computing a polynomial over matrices.

Forbes et al. [9] gave a $d^{O(\log w)}(nw)^{O(\log\log w)}$-time hitting-set for width-w, n-variate commutative ROABPs with individual degree bound d. Note that when d is small, this time complexity is much better than that for general ROABP, i.e., $(ndw)^{O(\log n)}$ [2]. However when d is $O(n)$, the complexity is comparable to the general case. We improve the time complexity for the commutative case to $(ndw)^{O(\log\log w)}$. This is significantly better than the general case for all values of d.

Forbes et al. [9] constructed the hitting-set using the notion of rank-concentration defined by Agrawal et al. [3].

▶ **Definition 4.1** ([3]). A polynomial $D(\boldsymbol{x})$ over an algebra is said to be ℓ-concentrated if its coefficients of $(<\ell)$-support monomials span all its coefficients.

Note that for a polynomial in $\mathbb{F}[\boldsymbol{x}]$, ℓ-concentration simply means that it has a monomial of $(<\ell)$-support with a nonzero coefficient. For a polynomial which has low-support concentration, it is easy to construct hitting-sets. However, not every polynomial has a low-support concentration, for example $C(\boldsymbol{x}) = x_1x_2\cdots x_n$. Agrawal et al. [3] observed that concentration can be achieved by a shift of variables, e.g., $C(\boldsymbol{x}+\mathbf{1}) = (x_1+1)(x_2+1)\cdots(x_n+1)$ has 1-concentration. For a polynomial $C(\boldsymbol{x})$, shift by a tuple $\boldsymbol{f} = (f_1, f_2, \ldots, f_n)$ would mean $C(\boldsymbol{x}+\boldsymbol{f}) = C(x_1+f_1, x_2+f_2, \ldots, x_n+f_n)$. The first step of Forbes et al. [9] is to show that for a given commutative width-w ROABP, $O(\log w)$-concentration can be achieved by a shift with cost $nd^{O(\log w)}$. Their second step is to show that if a given commutative ROABP is $O(\log w)$-concentrated then there is a hitting-set for it of size $(ndw)^{O(\log\log w)}$. We improve the first step by giving a shift with cost $(ndw)^{O(\log\log w)}$, which gives us the desired hitting-set.

First, we elaborate the first step of Forbes, Saptharishi and Shpilka [9]. To achieve concentration they use the idea of Agrawal, Saha and Saxena [3], i.e., achieving concentration in small sub-circuits implies concentration in the whole circuit. For the sake of completeness, we rewrite the lemma using the terminology of this paper.

▶ **Lemma 4.2** ([3, 9]). *Let $D(\boldsymbol{x}) = D_1(x_1)D_2(x_2)\cdots D_n(x_n)$ be a product of univariate polynomials over a commutative algebra $\mathbb{A}_k$. Suppose there exists an ℓ such that for any $S \in [n]$ with $|S| = \ell$, the polynomial $\prod_{i\in S} D_i$ has ℓ-concentration. Then $D(\boldsymbol{x})$ has ℓ-concentration.*

Proof. For any set $S \subseteq [n]$, let us define a sub-circuit D_S of D as $\prod_{i\in S} D_i(x_i)$. We will show ℓ-concentration in all the sub-circuits D_S of D, using induction on the size of S.

Base Case: D_S is trivially ℓ-concentrated if $|S| < \ell$. In the case of $|S| = \ell$, D_S is ℓ-concentrated from the hypothesis in the lemma.

Induction Hypothesis: D_S has ℓ-concentration for any set S with $|S| < j$.

Induction Step: We will prove ℓ-concentration in D_S for a set S with $|S| = j$. Let $S = \{x_{i_1}, x_{i_2}, \ldots, x_{i_j}\}$. Consider a monomial $\boldsymbol{x}^{\boldsymbol{a}} = x_{i_1}^{a_1}x_{i_2}^{a_2}\cdots x_{i_j}^{a_j}$ with support from the set S. Without loss of generality let us assume $a_1 \neq 0$. Now, let the set $S' = S \setminus \{x_{i_1}\}$ and let the monomial $\boldsymbol{x}^{\boldsymbol{a}'} = \boldsymbol{x}^{\boldsymbol{a}}/x_{i_1}^{a_1}$. As $|S'| = j-1$, by the inductive hypothesis $D_{S'}$ is ℓ-concentrated. Thus,

$$\operatorname{coef}_{D_{S'}}(\boldsymbol{x}^{\boldsymbol{a}'}) \in \operatorname{span}\{\operatorname{coef}_{D_{S'}}(\boldsymbol{x}^{\boldsymbol{b}}) \mid \operatorname{Supp}(\boldsymbol{b}) \subseteq S',\ \operatorname{supp}(\boldsymbol{b}) < \ell\}. \tag{1}$$

It is easy to see that for any monomial $\boldsymbol{x^b}$ with its support in S',

$$\operatorname{coef}_{D_S}(\boldsymbol{x^b} x_{i_1}^{a_1}) = \operatorname{coef}_{D_{S'}}(\boldsymbol{x^b}) \operatorname{coef}_{D_{i_1}}(x_{i_1}^{a_1}) .$$

Thus, by multiplying $\operatorname{coef}_{D_{i_1}}(x_{i_1}^{a_1})$ in (1), we get

$$\operatorname{coef}_{D_S}(\boldsymbol{x^a}) \in \operatorname{span}\{\operatorname{coef}_{D_S}(\boldsymbol{x^b} x_{i_1}^{a_1}) \mid \operatorname{Supp}(\boldsymbol{b}) \subseteq S', \operatorname{supp}(\boldsymbol{b}) < \ell\} .$$

Hence,

$$\operatorname{coef}_{D_S}(\boldsymbol{x^a}) \in \operatorname{span}\{\operatorname{coef}_{D_S}(\boldsymbol{x^b}) \mid \operatorname{Supp}(\boldsymbol{b}) \subseteq S, \operatorname{supp}(\boldsymbol{b}) \leq \ell\}. \tag{2}$$

Now, we claim that for any monomial $\boldsymbol{x^b}$ with $\operatorname{Supp}(\boldsymbol{b}) \subseteq S$ and $\operatorname{supp}(\boldsymbol{b}) = \ell$,

$$\operatorname{coef}_{D_S}(\boldsymbol{x^b}) \in \operatorname{span}\{\operatorname{coef}_{D_S}(\boldsymbol{x^c}) \mid \operatorname{Supp}(\boldsymbol{c}) \subseteq S, \operatorname{supp}(\boldsymbol{c}) < \ell\}. \tag{3}$$

To see this, let T be the support of the monomial $\boldsymbol{x^b}$. As $|T| = \ell$, D_T has ℓ-concentration. Thus,

$$\operatorname{coef}_{D_T}(\boldsymbol{x^b}) \in \operatorname{span}\{\operatorname{coef}_{D_T}(\boldsymbol{x^c}) \mid \operatorname{Supp}(\boldsymbol{c}) \subseteq T, \operatorname{supp}(\boldsymbol{c}) < \ell\}. \tag{4}$$

For any monomial $\boldsymbol{x^c}$ with support in T, one can write

$$\operatorname{coef}_{D_S}(\boldsymbol{x^c}) = \operatorname{coef}_{D_T}(\boldsymbol{x^c}) \prod_{i \in S \setminus T} \operatorname{coef}_{D_i}(1) .$$

Note that the commutativity of the underlying algebra is crucial for this. Thus, multiplying (4) by $\left(\prod_{i \in S\setminus T} \operatorname{coef}_{D_i}(1)\right)$, we get (3).

By combining (3) with (2), we get

$$\operatorname{coef}_{D_S}(\boldsymbol{x^a}) \in \operatorname{span}\{\operatorname{coef}_{D_S}(\boldsymbol{x^c}) \mid \operatorname{Supp}(\boldsymbol{c}) \subseteq S, \operatorname{supp}(\boldsymbol{c}) < \ell\} ,$$

for any monomial $\boldsymbol{x^a}$ with $\operatorname{Supp}(\boldsymbol{a}) \subseteq S$. This proves ℓ-concentration in D_S.

Taking $S = [n]$, we get ℓ-concentration in D. ◀

Now, the goal is just to achieve ℓ-concentration in an ℓ-variate ROABP (computing a polynomial over the matrix algebra). We would remark here that for an ℓ-variate polynomial over a k-dimensional algebra, one can hope to achieve ℓ-concentration only when $\ell \geq \log(k+1)$. To see this, consider the polynomial $D(\boldsymbol{x}) = \prod_{i=1}^{\ell}(1+v_i x_i)$ over a k-dimensional algebra such that $k > 2^\ell - 1$. Suppose the vector v_is are such that all the 2^ℓ coefficients of the polynomial D are linearly independent. There are only $2^\ell - 1$ coefficients of D with $(< \ell)$-support. Hence, they cannot span the whole coefficient space of D, whatever the shift we use.

Agrawal et al. [3] and Forbes et al. [9] achieve ℓ-concentration in arbitrary ℓ-variate polynomials over a k-dimension algebra for $\ell = \log(k+1)$ by a shift with cost $d^{O(\ell)}$, where d is the individual degree. Forbes et al. [9] use it to give a single shift on n variables such that it works for any choice of ℓ variables. This has cost $nd^{O(\ell)}$.

We give a new shift with cost $(ndw)^{O(\log \ell)} = (ndw)^{O(\log\log w)}$, for a width-$w$, ℓ-variate ROABP (w^2 is the dimension of the underlying algebra). The cost has n as a parameter because the shift works for any size ℓ subset of n variables. Like [3, 9], we use a shift by univariate polynomials in a new variable t. In this case, the concentration is considered over the field $\mathbb{F}(t)$. Note that while the shift of [3, 9] works for an arbitrary ℓ-variate polynomial, our shift works only for ℓ-variate ROABPs. The univariate map we use is the basis isolating weight assignment for ROABPs from Agrawal et al. [2]. We simply use the

fact that for any polynomial over a k-dimensional algebra, shift by a basis isolating map achieves $\log(k+1)$-concentration [11].

Let us first recall the definition of a basis isolating weight assignment. Let M denote the set of all monomials over the variable set $\boldsymbol{x}$ with individual degree $\leq d$. Any function $\mathrm{w}\colon \boldsymbol{x} \to \mathbb{N}$ can be naturally extended to the set of all monomials as follows: $\mathrm{w}(\prod_{i=1}^{n} x_i^{\gamma_i}) = \sum_{i=1}^{n} \gamma_i \mathrm{w}(x_i)$, for any $(\gamma_i)_{i=1}^{n} \in \mathbb{N}^n$. Note that if the variable x_i is replaced with $t^{\mathrm{w}(x_i)}$ for each i, then any monomial m just becomes $t^{\mathrm{w}(m)}$. $\mathbb{A}_k$ denotes a k-dimensional algebra.

▶ **Definition 4.3** ([2]). A weight function $\mathrm{w}\colon \boldsymbol{x} \to \mathbb{N}$ is called a basis isolating weight assignment for a polynomial $D(\boldsymbol{x}) \in \mathbb{A}_k[\boldsymbol{x}]$, if there exists a set of monomials $S \subseteq M$ ($k' := |S| \leq k$) whose coefficients form a basis for the coefficient space of $D(\boldsymbol{x})$, such that

- for any $m, m' \in S$, $\mathrm{w}(m) \neq \mathrm{w}(m')$ and
- for any monomial $m \in M \setminus S$,

$$\mathrm{coef}_D(m) \in \mathrm{span}\{\mathrm{coef}_D(m') \mid m' \in S,\ \mathrm{w}(m') < \mathrm{w}(m)\}\,.$$

Gurjar et al. [11, Lemma 5.2] have shown that shifting by a basis isolating weight assignment achieves concentration.

▶ **Lemma 4.4** (Isolation to concentration). *Let $A(\boldsymbol{x})$ be a polynomial over a k-dimensional algebra $\mathbb{A}_k$. Let* w *be a basis isolating weight assignment for $A(\boldsymbol{x})$. Then $A(\boldsymbol{x} + t^{\mathrm{w}})$ is ℓ-concentrated, where $\ell = \lceil \log(k+1) \rceil$ and t^{w} denotes the n-tuple $(t^{\mathrm{w}(x_1)}, t^{\mathrm{w}(x_2)}, \ldots, t^{\mathrm{w}(x_n)})$.*

We now recall the construction of a basis isolating weight assignment for ROABP from [2]. Here, we present a slightly modified version of their Lemma 8, which easily follows from it.

▶ **Lemma 4.5.** *Let $\boldsymbol{x}$ be a set of n variables. Let $D(\boldsymbol{x}) = D_1(x_{i_1})D_2(x_{i_2})\cdots D_\ell(x_{i_\ell})$ be an ℓ-variate polynomial over a k-dimensional algebra $\mathbb{A}_k$. Then we can construct a basis isolating weight assignment for $D(\boldsymbol{x})$ with the cost being $(\mathsf{poly}(k,n,d))^{\log \ell}$, where d is the individual degree.*

The construction in [2, Lemma 8] actually gives a family $\mathcal{B}$ of $(knd)^{O(\log \ell)}$ weight assignments such that for any ℓ-variate ROABP, at least one of them is basis isolating. However, we are interested in a single map which works for every ℓ-variate ROABP. To get a single shift for every ROABP, we follow the technique of [9, 11] and take a Lagrange Interpolation of all the n-tuples in the family $\{t^{\mathrm{w}}\}_{\mathrm{w}\in\mathcal{B}}$.

Let $\mathcal{F} = \{\boldsymbol{f}_1(t), \boldsymbol{f}_2(t), \ldots, \boldsymbol{f}_N(t)\}$ be this family of n-tuples, where $\boldsymbol{f}_i = \{f_{i,1}(t), f_{i,2}(t), \ldots, f_{i,n}(t)\}$ for each i. Here, $N = (knd)^{O(\log \ell)}$. Let their degrees be bounded by D, i.e., $D = \max\{\deg(f_{i,j}) \mid i \in [N] \text{ and } j \in [n]\}$. From the construction in [2], $D = (knd)^{O(\log \ell)}$. Also, the family $\mathcal{F}$ can be generated in time $(knd)^{O(\log \ell)}$.

Let $\boldsymbol{L}(y,t) \in \mathbb{F}[y,t]^n$ 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 $\{\alpha_i\}_{i\in[N]}$ are distinct field elements (we go to a large enough field extension where these many elements exist). Note that $L_j|_{y=\alpha_i} = f_{i,j}$. Thus, $\boldsymbol{L}|_{y=\alpha_i} = \boldsymbol{f}_i$. Also, $\deg_y(L_j) = N-1$ and $\deg_t(L_j) \leq D$. The following lemma from [11, Lemma 5.5] shows that a shift by the interpolation works for every polynomial simultaneously.

▶ **Lemma 4.6.** *Let $A(\boldsymbol{x})$ be a polynomial over $\mathbb{A}_k$ such that there exists an $\boldsymbol{f} \in \mathcal{F}$ for which $A'(\boldsymbol{x}, t) = A(\boldsymbol{x} + \boldsymbol{f}) \in \mathbb{A}_k(t)[\boldsymbol{x}]$ is ℓ-concentrated. Then, $A''(\boldsymbol{x}, y, t) = A(\boldsymbol{x} + \boldsymbol{L}) \in \mathbb{A}_k(y, t)[\boldsymbol{x}]$ is ℓ-concentrated.*

Proof. Let $\text{rank}_{\mathbb{F}}\{\text{coef}_A(\boldsymbol{x}^{\boldsymbol{a}}) \mid \boldsymbol{x}^{\boldsymbol{a}} \in M\} = k'$, for some $k' \leq k$, and $M_\ell = \{\boldsymbol{x}^{\boldsymbol{a}} \in M \mid \text{supp}(\boldsymbol{a}) < \ell\}$. We need to show that $\text{rank}_{\mathbb{F}(y,t)}\{\text{coef}_{A''}(\boldsymbol{x}^{\boldsymbol{a}}) \mid \boldsymbol{x}^{\boldsymbol{a}} \in M_\ell\} = k'$.

Since $A'(\boldsymbol{x})$ is ℓ-concentrated, we have that $\text{rank}_{\mathbb{F}(t)}\{\text{coef}_{A'}(\boldsymbol{x}^{\boldsymbol{a}}) \mid \boldsymbol{x}^{\boldsymbol{a}} \in M_\ell\} = k'$. Recall that $A'(\boldsymbol{x})$ is an evaluation of A'' at $y = \alpha_i$, i.e., $A'(\boldsymbol{x}, t) = A''(\boldsymbol{x}, \alpha_i, t)$ for some α_i. Thus, for all $\boldsymbol{x}^{\boldsymbol{a}} \in M$, we have $\text{coef}_{A'}(\boldsymbol{x}^{\boldsymbol{a}}) = \text{coef}_{A''}(\boldsymbol{x}^{\boldsymbol{a}})|_{y=\alpha_i}$.

Let $C \in \mathbb{F}[t]^{k \times |M_\ell|}$ be the matrix whose columns are $\text{coef}_{A'}(\boldsymbol{x}^{\boldsymbol{a}})$, for $\boldsymbol{x}^{\boldsymbol{a}} \in M_\ell$. Let similarly $C' \in \mathbb{F}[y, t]^{k \times |M_\ell|}$ be the matrix whose columns are $\text{coef}_{A''}(\boldsymbol{x}^{\boldsymbol{a}})$, for $\boldsymbol{x}^{\boldsymbol{a}} \in M_\ell$. Then we have $C = C'|_{y=\alpha_i}$.

As $\text{rank}_{\mathbb{F}(t)}(C) = k'$, there is a $k' \times k'$ submatrix in C, say indexed by (R, T), such that $\det(C(R, T)) \neq 0$. Since $\det(C(R, T)) = \det(C'(R, T))|_{y=\alpha_i}$, it follows that $\det(C'(R, T)) \neq 0$. Hence, we have $\text{rank}_{\mathbb{F}(y,t)}(C') = k'$. Thus, the $(< \ell)$-support coefficients of A'' span its coefficient space. ◀

Hence, the Lagrange interpolation gives us a single shift which works for all ℓ-variate ROABPs.

▶ **Lemma 4.7.** *Given n, d, w and $\ell = \log(w^2 + 1)$, in time $(ndw)^{O(\log \ell)}$ one can compute a polynomial tuple $\boldsymbol{f}(t) \in \mathbb{F}[t]^n$ of degree $(ndw)^{O(\log \ell)}$ such that for any ℓ-variate polynomial $A(\boldsymbol{x}) \in \mathbb{F}^{w \times w}[\boldsymbol{x}]$ of individual degree d that can be computed by an ROABP of width w, the polynomial $A(\boldsymbol{x} + \boldsymbol{f}(t))$ is ℓ-concentrated.*

Proof. Note that the dimension k of the underlying algebra is bounded by w^2. After shifting the polynomial $A(\boldsymbol{x})$ by $\boldsymbol{L}(y, t)$ as defined above, its coefficients will be polynomials in y and t, with degree $d' = dn(ndw)^{O(\log \ell)}$. Consider the determinant polynomial $\det(C'(R, T))$ from the proof of Lemma 4.6. As $k' \leq k$, $\det(C'(R, T))$ has degree bounded by $d'' := kd'$. So, when we replace y by $t^{d''+1}$, it does not affect the non-zeroness of the determinant, and hence, the concentration is preserved. Thus, $\boldsymbol{f} = \boldsymbol{L}(t^{d''+1}, t)$ is an n-tuple of univariate polynomials in t that fulfils the claim of the lemma. ◀

Combining Lemma 4.2 and Lemma 4.7 we get the following.

▶ **Lemma 4.8.** *Given n, d, w, one can compute an n-tuple $\boldsymbol{f}(t)$ with cost $(ndw)^{O(\log \log w)}$ such that for any n-variate, individual degree-d polynomial $D(\boldsymbol{x}) \in \mathbb{F}^{w \times w}[\boldsymbol{x}]$ computed by a width-w commutative ROABP, $D(\boldsymbol{x} + \boldsymbol{f}(t))$ is $O(\log w)$-concentrated.*

Note that if the polynomial $D(\boldsymbol{x}) \in \mathbb{F}^{w \times w}[\boldsymbol{x}]$ is ℓ-concentrated then the polynomial $C(\boldsymbol{x}) = U^{\mathsf{T}} D T$ is also ℓ-concentrated, where $U, T \in \mathbb{F}^{w \times 1}$. This is true because multiplication by U^{T} and T are linear operations. Recall that for polynomial $C(\boldsymbol{x}) \in \mathbb{F}[\boldsymbol{x}]$, $O(\log w)$-concentration means that there is a monomial with $O(\log w)$-support which has a nonzero coefficient.

Lemma 4.8 gives a shift $\boldsymbol{f}(t)$ of univariate polynomials. To get a constant shift, we substitute $(ndw)^{O(\log \log w)}$ distinct values for t. As the degree in t is bounded by $(ndw)^{O(\log \log w)}$, at least for one value of t, the non-zeroness of the particular coefficient will be preserved.

Now, we move on to the second step of Forbes, Shpilka and Saptharishi [9]. They give an $(ndw)^{O(\log \log w)}$-time hitting-set for an already $O(\log w)$-concentrated commutative ROABP. They do this by reducing the PIT question to an $O(\log w)$-variate ROABP [9, Lemma 7.6].

▶ **Lemma 4.9** ([9]). *Let $C(\boldsymbol{x}) \in \mathbb{F}[\boldsymbol{x}]$ be an n-variate, individual degree-d polynomial computed by a width-w ROABP. Suppose $C(\boldsymbol{x})$ has an $(\leq \ell)$-support monomial with a nonzero coefficient. Then, there is a $\mathsf{poly}(n, w, d)$-time computable m-variate map $\phi\colon \boldsymbol{x} \to \mathbb{F}[y_1, y_2, \ldots, y_m]$ such that $C(\phi(\boldsymbol{x}))$ is a nonzero polynomial with degree $< d^2n^4$, where $m = O(\ell^2)$. Moreover, $C(\phi(\boldsymbol{x}))$ is computed by a width-w, m-variate commutative ROABP.*

From the results of [10, 2], we know that an m-variate, width-w commutative ROABP has an $(mdw)^{O(\log m)}$-time hitting-set. Combining Lemma 4.8 and Lemma 4.9 with this fact and putting $m = O(\log^2 w)$, we get the following.

▶ **Theorem 4.10.** *There is an $(ndw)^{O(\log\log w)}$-time hitting-set for n-variate commutative ROABPs with width w and individual degree d.*

Concentration in Set-multilinear Circuits: Similar to Theorem 4.10, it would be interesting to achieve the same time complexity for set-multilinear circuits. Recall from Section 2.2.3 that a polynomial computed by a depth-3 set-multilinear circuit can be written as $(1, 1, \ldots, 1) \cdot D$, where $D = D_1(\boldsymbol{x}_1)D_2(\boldsymbol{x}_2)\cdots D_q(\boldsymbol{x}_q)$ is a product of linear polynomials over a commutative algebra. It is easy to see that the same arguments as for commutative ROABP work here. Hence, we get the following result analogous to Lemma 4.8.

▶ **Corollary 4.11.** *Given n, k, one can compute an n-tuple $\boldsymbol{f}(t)$ with cost $(nk)^{O(\log\log k)}$ such that for any n-variate polynomial $C(\boldsymbol{x})$ computed by a depth-3 set-multilinear circuit with top fan-in k, $C(\boldsymbol{x} + \boldsymbol{f}(t))$ is $O(\log k)$-concentrated.*

However, it is not clear whether the second step of the hitting-set construction can be done for set-multilinear circuits, i.e., finding a better hitting-set by assuming that the polynomial is already concentrated (Lemma 4.9).

5 Discussion

For our first result (Theorem 3.6), there are three directions of improvement. Ideally, one would like to have all three at once.

1. Find a similar hitting-set for the unknown-order case. In fact, we conjecture that the same hitting-set (Lemma 3.5) works for the unknown-order case as well.
2. Get a hitting-set for all characteristic fields. It is easy to construct examples over small characteristic fields where our hitting-set does not work.
3. Reduce the time complexity to polynomial time. To achieve this, it seems one has to do away with the divide and conquer approach.

We conjecture a polynomial-time hitting-set for the unknown-order case in Conjecture 3.8.

As mentioned earlier, the ideas here can help in finding a better PRG for ROBPs. In particular, it is a big open question to find an $O(\log n)$-seed-length PRG for constant-width ROBPs (analogous to Corollary 3.7).

References

1 Manindra Agrawal. Proving lower bounds via pseudo-random generators. In *FSTTCS*, volume 3821 of *Lecture Notes in Computer Science*, pages 92–105, 2005.

2 Manindra Agrawal, Rohit Gurjar, Arpita Korwar, and Nitin Saxena. Hitting-sets for ROABP and sum of set-multilinear circuits. *SIAM J. Comput.*, 44(3):669–697, 2015. doi:10.1137/140975103.

3 Manindra Agrawal, Chandan Saha, and Nitin Saxena. Quasi-polynomial hitting-set for set-depth-formulas. In *STOC*, pages 321–330, 2013.

4 Matthew Anderson, Michael Forbes, Ramprasad Saptharishi, Amir Shpilka, and Ben Lee Volk. Identity testing and lower bounds for read-k oblivious algebraic branching programs. In *Computational Complexity Conference (CCC)*, 2016.

5 Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, New York, NY, USA, 1st edition, 2009.

6 Michael Ben-Or and Richard Cleve. Computing algebraic formulas using a constant number of registers. *SIAM J. Comput.*, 21(1):54–58, 1992.

7 Rafael Mendes de Oliveira, Amir Shpilka, and Ben Lee Volk. Subexponential size hitting sets for bounded depth multilinear formulas. In *30th Conference on Computational Complexity (CCC)*, pages 304–322, 2015. doi:10.4230/LIPIcs.CCC.2015.304.

8 Richard A. Demillo and Richard J. Lipton. A probabilistic remark on algebraic program testing. *Information Processing Letters*, 7(4):193–195, 1978.

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), New York, NY, USA, May 31 – June 03, 2014*, pages 867–875, 2014.

10 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. doi:10.1109/FOCS.2013.34.

11 Rohit Gurjar, Arpita Korwar, Nitin Saxena, and Thomas Thierauf. Deterministic identity testing for sum of read-once oblivious arithmetic branching programs. In *30th Conference on Computational Complexity, CCC 2015, June 17-19, 2015, Portland, Oregon, USA*, pages 323–346, 2015. doi:10.4230/LIPIcs.CCC.2015.323.

12 J. Heintz and C. P. Schnorr. Testing polynomials which are easy to compute (extended abstract). In *Proceedings of the Twelfth Annual ACM Symposium on Theory of Computing*, STOC'80, pages 262–272, New York, NY, USA, 1980. ACM. doi:10.1145/800141.804674.

13 Russell Impagliazzo, Raghu Meka, and David Zuckerman. Pseudorandomness from shrinkage. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 111–119, 2012. doi:10.1109/FOCS.2012.78.

14 Russell Impagliazzo, Noam Nisan, and Avi Wigderson. Pseudorandomness for network algorithms. In *Proceedings of the Twenty-sixth Annual ACM Symposium on Theory of Computing*, STOC, pages 356–364, New York, NY, USA, 1994. ACM.

15 Valentine Kabanets and Russell Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *STOC*, pages 355–364, 2003.

16 Neeraj Kayal, Vineet Nair, and Chandan Saha. Separation between read-once oblivious algebraic branching programs (roabps) and multilinear depth three circuits. In *33rd Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 46:1–46:15, 2016.

17 Pascal Koiran. Arithmetic circuits: The chasm at depth four gets wider. *Theoretical Computer Science*, 448:56–65, 2012. doi:10.1016/j.tcs.2012.03.041.

18 Noam Nisan. Pseudorandom generators for space-bounded computations. In *Proceedings of the Twenty-second Annual ACM Symposium on Theory of Computing*, STOC'90, pages 204–212, New York, NY, USA, 1990. ACM. doi:10.1145/100216.100242.

19 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.

20 Ran Raz and Omer Reingold. On recycling the randomness of states in space bounded computation. In *Proceedings of the Thirty-First Annual ACM Symposium on Theory of*

Computing, May 1-4, 1999, Atlanta, Georgia, USA, pages 159–168, 1999. doi:10.1145/301250.301294.

21 Ran Raz and Amir Shpilka. Deterministic polynomial identity testing in non-commutative models. *Computational Complexity*, 14(1):1–19, 2005.

22 Jacob T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *J. ACM*, 27(4):701–717, October 1980.

23 Richard Zippel. Probabilistic algorithms for sparse polynomials. In *Proceedings of the International Symposiumon on Symbolic and Algebraic Computation*, EUROSAM'79, pages 216–226, London, UK, UK, 1979. Springer-Verlag.

Identity Testing and Lower Bounds for Read-k Oblivious Algebraic Branching Programs*

Matthew Anderson[1], Michael A. Forbes[2], Ramprasad Saptharishi[3], Amir Shpilka[4], and Ben Lee Volk[5]

1 **Department of Computer Science, Union College, Schenectady, USA**
andersm2@union.edu
2 **Department of Computer Science, Princeton University, Princeton, USA**
miforbes@csail.mit.edu
3 **Department of Computer Science, Tel Aviv University, Tel Aviv, Israel**
ramprasad@cmi.ac.in
4 **Department of Computer Science, Tel Aviv University, Tel Aviv, Israel**
shpilka@post.tau.ac.il
5 **Department of Computer Science, Tel Aviv University, Tel Aviv, Israel**
benleevolk@gmail.com

Abstract

Read-k oblivious algebraic branching programs are a natural generalization of the well-studied model of read-once oblivious algebraic branching program (ROABPs). In this work, we give an exponential lower bound of $\exp(n/k^{O(k)})$ on the width of any read-k oblivious ABP computing some explicit multilinear polynomial f that is computed by a polynomial size depth-3 circuit. We also study the polynomial identity testing (PIT) problem for this model and obtain a white-box subexponential-time PIT algorithm. The algorithm runs in time $2^{\tilde{O}(n^{1-1/2^{k-1}})}$ and needs white box access only to know the order in which the variables appear in the ABP.

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

Keywords and phrases Algebraic Complexity, Lower Bounds, Derandomization, Polynomial Identity Testing

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.30

1 Introduction

Algebraic complexity studies the complexity of syntactically computing polynomials using arithmetic operations. The most natural model for computing polynomials is an *algebraic circuit*, which is a directed, acyclic graph whose leaves are labeled by either variables from $\{x_1, \ldots, x_n\}$ or elements from a field $\mathbb{F}$, and whose internal nodes use the arithmetic operations $+$ and $\times$. Each node thus computes a polynomial in the natural way. The associated complexity measures are the *size* (the number of wires) and the *depth* (the length of a longest path from an input node to the output node) of the circuit. A circuit whose underlying graph is a tree is called a *formula*.

* 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, and from the Princeton Center for Theoretical Computer Science.

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 30; pp. 30:1–30:25

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

Another model of computation, whose power lies between that of circuits and formulas, is that of an *algebraic branching program* (ABP). An ABP is a directed layered acyclic graph with a source node and a sink node, whose edges are labeled by polynomials. An ABP computes a polynomial in the following way. Every directed source-sink path computes the polynomial that is obtained from taking the product of all edge labels along the path. The polynomial computed by the ABP is the sum over all paths of those polynomials.[1] Here, another relevant complexity measure is the *width* of the program, which is the maximal number of vertices in a layer (see Section 1.1 for the exact definitions of the models that are considered in this work).

Two of the most important problems in algebraic complexity are (i) proving exponential lower bounds for arithmetic circuits (i.e., proving that any circuit computing some explicit polynomial f must be of exponential size), and (ii) giving an efficient *deterministic* algorithm for the *polynomial identity testing* (PIT) problem. The latter is the problem of given an arithmetic circuit, formula or ABP, computing a polynomial f, we have to decide whether f is the identically zero polynomial. PIT has a simple randomized algorithm that follows from the Schwartz-Zippel-DeMillo-Lipton lemma [59, 70, 17] that says that over a large enough field, a non-zero polynomial will evaluate to a non-zero value on *most* points. Hence, in order to decide whether f is zero it is enough to evaluate the circuit/formula/ABP on a random point (which can be done efficiently).

We further note that the randomized algorithm described above only needs the ability to evaluate f at a given point. Such algorithms are called *black-box* PIT algorithms. It is readily seen that black-box algorithms are equivalent to producing a small *hitting set*, which is a set $\mathcal{H}$ of evaluation points that has the property that $\mathcal{H}$ contains a non-zero evaluation point for every non-zero f. Algorithms that are given the computation graph as input are called *white-box* algorithms. Naturally, white-box access is much less restrictive and one expects it will be easier to obtain better algorithms in this case.

Apart from being a very natural problem about arithmetic computation, PIT is one of the most general problems for which an efficient randomized algorithm is known, but no deterministic one. Indeed, many other randomized algorithms — e.g. parallel algorithms for finding matching in graphs [36, 49] or algorithms for polynomial factorization [60, 43] — reduce to PIT, in the sense that derandomization of PIT would derandomize those as well.

For more background on arithmetic circuits we refer the reader to the survey [62].

At first glance, the two problems described above seem rather different, as one is concerned with proving lower bounds and the other with providing efficient algorithms. However, a series of works uncovered an intricate web of connections between the two, both in the white-box [34, 19] and in the black-box [30, 1] models. That is, derandomizing PIT implies lower bounds for circuits (which gives a convincing explanation for why this problem is hard), and conversely, an explicit hard polynomial gives a recipe to "fool" small arithmetic circuits with respect to non-zeroness, in a very similar manner to the hardness-versus-randomness paradigm in boolean complexity.

In light of the hardness of proving lower bounds for general circuits, research has focused on trying to understand the effect that structural restrictions, like constant depth and multilinearity, have on the expressive power of the model.

One research direction that has attracted a lot of attention considers very shallow depth arithmetic circuits. Following Valiant et al. [67], Agrawal and Vinay gave a reduction from

[1] This is analogous to boolean branching programs. There each path computes the AND of edge labels and the output is the OR of all path-functions.

general circuits (whose degree is at most the number of variables) to depth-4 circuits, that maps subexponential size to subexponential size [3]. This reduction was later improved and extended in [42, 65, 27]. In a breakthrough work Gupta et al. [28] proved exponential lower bounds for computing the $n \times n$ determinant by depth-4 homogeneous formulas with bottom fan-in $O(\sqrt{n})$. This is the kind of circuits one gets from the depth reduction. In the works subsequent to [28], tighter lower bounds for homogeneous depth-4 circuits were proved both for "hard" polynomials such as the permanent but also for easier polynomials such as the determinant and the iterated matrix multiplication polynomial [39, 25, 37, 47, 46].

In parallel, a lot of research effort was also focused on PIT for small-depth circuits with various restrictions such as bounded top fan-in or multilinearity [18, 41, 40, 58, 35, 57, 53]. Similar to the situation with lower bounds, a derandomization of PIT for depth-4 circuits (or, depth-3 in certain cases) implies a derandomization of the general case [3, 27]. Shpilka and Volkovich [61] studied the class of sums of read-once arithmetic formulas (here, a read-once formula is an arithmetic formula in which each variable labels at most one node) and gave polynomial identity tests for this model [61]. Later, Anderson, van Melkebeek and Volkovich gave a PIT for multilinear read-k formulas [8].

Another line of work focused on read-once oblivious ABPs (ROABPs, and we again refer to Section 1.1 for the exact definition). ROABPs were defined by Nisan [50] in the context of proving lower bounds for non-commutative formulas. While this model seems a bit restrictive, it was shown that derandomizing PIT for ROABPs implies derandomization of Noether's normalization lemma for certain important varieties [48, 23]. It is also not hard to show that ROABPs are strictly stronger than read-once arithmetic formulas. Another motivation to study this model is that it is the algebraic analog of a boolean read-once branching program, which arises in the context of pseudorandomness for small-space computation [51]. Thus, one could hope for cross-fertilization of ideas between the models that could facilitate progress on both fronts.

Exponential lower bounds for ROABPs were known since their inception [50], and a white-box polynomial-time PIT algorithm was given by Raz and Shpilka [54]. In the black-box setting, hitting sets of quasipolynomial size were obtained in [24, 22, 2], where the last two papers being applicable even if the order in which the variable are read is unknown (the hitting of [22] was quasipolynomial sized for *bounded individual degree*, but the subsequent hitting set of [2] is quasipolynomial sized for any $d = \text{poly}(n)$). This marks a striking difference between the algebraic model and the boolean model. Indeed, in the boolean domain, pseudorandom generators for read-once branching programs in unknown order are much weaker, in terms of the seed length, than Nisan's generator [51] which works only if the order is known. Recently, Gurjar et al. obtained PIT algorithms for sum of ROABPs [29].

In this work, we consider the natural next step, which are read-k oblivious algebraic branching programs. This model generalizes and extends both the models of ROABPs, of read-k arithmetic formulas and of sum of ROABPs. We are able to prove exponential lower bounds and to give subexponential-time PIT algorithms for this model. A summary of our results appears in Section 1.2.

Prior to our work there were no results known for this model. Some results were known for the more restricted model of a *sum* of k ROABPs (e.g. [29]), and we give more details on those in Section 1.3.

1.1 Computational Models

In this section we define the computational models we consider in this work. We begin with the definition of *Algebraic Branching Programs* (ABPs).

▶ **Definition 1.1** (Algebraic Branching Program, [50])**.** 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 layers labeled $0, 1, \ldots, L$. Edges in the graph can only go from layer $\ell - 1$ to layer ℓ, for $\ell \in [L]$. The source is the only vertex at layer 0 and the sink is the only vertex at layer L. Each edge is labeled with a polynomial 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. The *degree* of an ABP is defined to be the maximal degree of the polynomial edge labels.

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 to t paths, of such polynomials.

The expressive power of ABPs lies between arithmetic formulas and arithmetic circuits. Every formula of size s can be simulated by an ABP of size s. Similarly, an ABP of width s and depth d can be simulated by an arithmetic circuit of size $O(sd^2)$.

In this work we consider a restricted model of ABPs that we call read-k oblivious ABPs. In an oblivious ABP, in each layer all the labels are univariate polynomials in the same variable. Furthermore, we also restrict each variable to appear in at most k layers while still allowing them to label any number of the edges in those layers.

▶ **Definition 1.2** (Read-k Oblivious ABPs, [24])**.** An algebraic branching program is said to be *oblivious* if for every layer ℓ, all the edge labels in that layer are univariate polynomials in a variable x_{i_ℓ}.

Such a branching program is said to be a *read-once* oblivious ABP (ROABP) if the x_{i_ℓ}'s are distinct variables. That is, each x_i appears in the edge labels in at most one layer.

An oblivious ABP is said to be a *read-k* if each variable x_i appears in the edge labels of at most k layers.

▶ nremark 1.3. For the rest of the discussion, it will be convenient to assume that in a read-k oblivious ABP, every variable x appears in *exactly* k layers. This assumption can be made without loss of generality, since if x appears in $k' < k$ layers, we can add $k - k'$ "identity" layers to the program that vacuously read x. This transformation does not increase the width of the program and increases the length by no more than kn.

A special case of a read-k oblivious ABP is one where the ABP makes "multiple passes" over the input.

▶ **Definition 1.4** (k-pass ABPs)**.** An oblivious ABP is said to be a *k-pass ABP* if there exists a permutation π on n such that the ABP reads variables in the order

$$\underbrace{x_{\pi(1)}, \ldots, x_{\pi(n)}, x_{\pi(1)}, \ldots, x_{\pi(n)}, \ldots, x_{\pi(1)}, \ldots, x_{\pi(n)}}_{k \text{ times}}.$$

An oblivious ABP is said to be a *k-pass varying-order ABP* if there are permutations $\pi_1, \cdots, \pi_k$ over n symbols such that the ABP reads variables in the order

$$x_{\pi_1(1)}, \ldots, x_{\pi_1(n)}, x_{\pi_2(1)}, \ldots, x_{\pi_2(n)}, \ldots, x_{\pi_k(1)}, \ldots, x_{\pi_k(n)}.$$

1.2 Our Results

We give various results about the class of read-k oblivious ABPs, including lower bounds, PIT algorithms, and separations.

1.2.1 Lower Bounds

We show an explicit polynomial f such that any read-k oblivious ABP computing f, for bounded k, must be of exponential width.

▶ **Theorem 1.5** (proved in Section 4). *There exists an explicit polynomial f, which is computed by a depth-3 polynomial-size multilinear circuit, such that any read-k oblivious ABP computing f must have width $\exp(n/k^{O(k)})$.*

Prior to this work, there were no lower bounds for this model.

1.2.2 Identity Testing

For the class of k-pass ABPs, we provide a black-box PIT algorithm that runs in quasipolynomial time.

▶ **Theorem 1.6** (proved in Subsection 5.1). *There exists a black-box PIT algorithm for the class of n-variate, degree-d, and width-w k-pass oblivious ABPs that runs in time $(nw^{2k}d)^{O(\log n)}$.*

For the more general class of read-k oblivious ABPs, we provide a white-box PIT algorithm that runs in subexponential time.

▶ **Theorem 1.7** (proved in Section 5). *There exists a white-box PIT algorithm for the class of n-variate, degree-d, and width-w read-k oblivious ABPs that runs in time $(nwd)^{\tilde{O}(n^{1-1/2^{k-1}})\cdot\exp(k^2)}$. Furthermore, white-box access is only needed to know the order in which the variables are read. That is, given this order, we construct an explicit hitting set of the above size for the class of read-k oblivious ABPs that read their variables in that order.*

1.2.3 Separations

Recently, Kayal, Nair and Saha [38] constructed a polynomial f that can be computed by a sum of *two* ROABPs in different orders, each of constant width, such that any ROABP computing f must be of width $2^{\Omega(n)}$. Note that sum of two ROABPs is a special case of a 2-pass varying-order ABP.

In order to exemplify the strength of the multiple-reads model, we show a polynomial that can be computed by a small 2-pass varying-order ABP, but cannot be computed by a small sum of ROABPs of small width.

▶ **Theorem 1.8** (proved in Section 3). *There exists an explicit polynomial f on n^2 variables that is computed by a 2-pass varying-order ABP of constant width, but any sum of c ROABPs computing f must be of width $\exp(\Omega(\sqrt{n}/2^c))$.*

1.3 Related Work

1.3.1 Algebraic Models

As mentioned before, Nisan [50] proved exponential lower bounds for ROABPs, and Raz and Shpilka [54] gave a white-box polynomial-time PIT algorithm for this model.

Forbes and Shpilka [24] were the first to consider the black-box version of this problem, and obtained a hitting set of size $(nwd)^{O(\log n)}$, for n-variate, degree-d and width-w ROABPs, if the order in which the variables are read is known in advance. Forbes, Shpilka and Saptharishi [22] obtained a hitting set of size $(nwd)^{O(d\log(w)\log n)}$ for *un*known order ROABPs. This was

improved later by Agrawal et al. [2] who obtained a hitting set of size $(nwd)^{O(\log n)}$ which matches the parameters of the known-order case.

For higher number of reads, much less was known. Gurjar et al. [29] considered the model of a *sum* of c ROABPs, and obtained a white-box algorithm that runs in time $(ndw^{2^c})^{O(c)}$, and a black-box algorithm that runs in time $(ndw)^{O(c2^c \log(ndw))}$, so that the running time is polynomial in the former case and quasipolynomial in the latter, when c is constant. A sum of c ROABPs can be simulated by read-c oblivious ABPs, and we show (in Section 3) that read-c oblivious ABPs are in fact strictly stronger.

Lower bounds against the model of sums of ROABPs were obtained in a recent work of Arvind and Raja [9], who showed that for every constant $\varepsilon > 0$, if the permanent is computed by a sum of $n^{1/2-\varepsilon}$ ROABPs, then at least one of the ROABPs must be of width $2^{n^{\Omega(1)}}$.

We also mention an earlier work of Jansen et al. [33], who also gave white-box and black-box tests for the weaker model of sum of constantly many read-once ABPs, where in their definition every variable is allowed to label only a single *edge* in the ABP.

Another model which is subsumed by oblivious read-k ABPs is that of bounded-read formulas. Shpilka and Volkovich [61] constructed quasipolynomial-size hitting set for read-once formulas, and Anderson, van Melkebeek and Volkovich [8] extended this result to multilinear read-k formulas and obtained a polynomial-time white-box algorithm and quasipolynomial-time black-box algorithm. The natural simulation of read-k formulas by ABPs produces an ABP in which every variable labels at most k edges, and it can be seen that such programs can be converted to read-k oblivious ABPs with only a polynomial overhead.

To conclude, earlier results apply only to restricted submodels of read-k oblivious ABPs.

1.3.2 Boolean Models

Let us now make a small detour and consider the boolean analogs for our models. A (boolean) branching program is a directed acyclic graph with a source node s and two sink nodes, t_0 and t_1. Each internal *node* is labeled by a variable x_i with two outgoing edges, labeled 0 and 1. The program computes a boolean function on an input $(x_1, \ldots, x_n) \in \{0,1\}^n$ by following the corresponding path along the program.

A read-k-times boolean branching program is allowed to query every variable at most k times along every path from the source to the sink. Note that this is more general than our definition of read-k oblivious branching program. Further distinction is made in the boolean case between *semantic* read-k branching programs, in which this restriction is enforced only on paths that are consistent with some input, and between *syntactic* read-k branching programs, in which this restriction applies for all paths (further note that in the read-once case, there is no distinction between the syntactic and the semantic model).

Exponential lower bounds for read-once branching program for explicit functions are known since the 1980's [69, 10, 68], even for functions that are computed by a polynomial size read-twice branching program.

Okolnishnikova [52], and Borodin, Razborov and Smolensky [14] extended these results and obtained exponential lower bounds for syntactic read-k-times branching programs, by giving an explicit boolean function f such that every syntactic read-k-times branching program for f has size $\exp(n/2^{O(k)})$ (in fact, the lower bound in the second work also holds for the stronger class of non-deterministic branching programs).

A strong separation result was obtain by Thathachar [66], who showed a *hierarchy theorem* for syntactic read-k-times boolean branching program, by giving, for every k, a boolean function f which is computed by a linear-size syntactic read-$(k+1)$-times branching

program such that every syntactic read-k-times branching program computing f must have size $\exp(\Omega(n^{1/k}/2^{O(k)}))$.

The semantic model seemed more difficult, but nevertheless Ajtai [5] was able to prove an exponential lower bound for semantic read-k-times programs (when k is constant), which was extended by Beame at al. [11] to randomized branching programs.

PIT is the algebraic analog of constructing pseudorandom generators (PRGs) for boolean models. A PRG for a class $\mathcal{C}$ of boolean circuits is an easily computable function $G : \{0,1\}^{\ell} \to \{0,1\}^{n}$, such that for any circuit $C \in \mathcal{C}$, the probability distributions $C(U_n)$ and $C(G(U_\ell))$ are ε-close (where U_m is the uniform distribution over $\{0,1\}^m$).

Nisan [51] constructed a PRG for polynomial size read-once oblivious branching programs with seed length $O(\log^2 n)$. This was followed by a different construction with the same seed length by Impagliazzo, Nisan and Wigderson [32]. However, for the constructions to work it is crucial that the order in which the variables are read is known in advance.

Beyond that, and despite a large body of work devoted to this topic [12, 13, 15, 16, 26, 31, 44, 56, 63, 64], all the results for the unknown order case or for read-k oblivious branching programs have much larger seed length, unless further structural restrictions are put on the program (such as very small width, regularity, or being a permutation branching programs). Specifically, we highlight that even for read-2 oblivious branching programs, the best result is by Impagliazzo, Meka and Zuckerman [31] who gave a PRG with seed length $s^{1/2+o(1)}$ for size s branching program (note that the the dependence here is on s rather than on n). In particular, no non-trivial results are known for general polynomial size read-2 oblivious boolean branching program.

1.4 Proof Technique

Before delving into the details of our proof, it is perhaps instructive to think again about read-once branching programs. The main exploitable weakness of these branching programs is that by the read-once property, their computation can be broken into two subcomputations over disjoint variables, that communicate with each other only through a small "window" of width w, the width of the branching program. If w is small it is natural to expect that upon reaching the middle layer, the branching program must "forget" most of the computation of the first half so that both subcomputations are "almost independent" in a way. This property calls for a divide-and-conquer strategy, which was indeed, in very crude terms, the strategy that was applied both in the boolean model [51] and in the algebraic model [24, 22, 2] (the details in each case, of course, are much more complicated than this simplistic description).

1.4.1 Evaluation dimension and ROABPs

Unfortunately, the above intuition breaks down when we allow a variable to be read multiple times, and this model requires a different strategy. Our main starting point is the observation that, perhaps surprisingly, multiple "passes" over the input variables, *in the same order*, do not provide the program with much additional power. That is, a k-pass ABP can be simulated by a ROABP, with a blow-up which is exponential in k (hence, only a polynomial blow-up, if k is constant).

This fact can be directly seen through analysis of the *evaluation dimension* measure. For a polynomial $f(x_1, \ldots, x_n) \in \mathbb{F}[x_1, \ldots, x_n]$ and a subset of variables S, we denote by $\text{eval}_S(f)$ the subspace of $\mathbb{F}[x_1, \ldots, x_n]$ that consists of all the possible polynomials obtained from f by fixing the variables in S to arbitrary elements in $\mathbb{F}$. The evaluation dimension of f with respect to a partition $S, \overline{S}$, which is denoted $\text{evalDim}_{S,\overline{S}}(f)$ is the dimension of

$\text{eval}_S(f)$. Over large enough fields, this dimension equals the rank of the *partial derivative matrix* associated with this partition, as defined by Nisan [50]. In many contexts, however, it is easier to work with the evaluation dimension. We refer to Chapter 4 of [21] for a detailed discussion on this equivalence, including formal proofs.

The importance of the evaluation dimension measure stems from the fact that f can be computed by a width-w ROABP in the order $x_1, x_2, \ldots, x_n$, if and only if $\text{evalDim}_{\{x_1,\ldots,x_i\},\{x_{i+1},\ldots,x_n\}}(f) \leq w$ for every $1 \leq i \leq n$. Thus, this measure provides a precise characterization for the amount of resources needed to compute a polynomial in this model (see Theorem 2.4).

1.4.2 Evaluation dimension and k-pass oblivious ABPs

We are able to adapt the proof of the "only if" part of the above fact in order to show that if f is computed by a k-pass oblivious ABP (that is, f reads the n variables k times in the same order) then $\text{evalDim}_{\{x_1,\ldots,x_i\},\{x_{i+1},\ldots,x_n\}}(f) \leq w^{2k}$ for every $i \in [n]$. That is, k passes over the input in the same order cannot create many independent evaluations. Then, using the "if" part of the equivalence, it follows that f can also be computed using a ROABP of width w^{2k} (see Lemma 2.6).

This discussion immediately implies a hitting set of the class of k-pass oblivious ABPs of size $(ndw^{2k})^{O(\log n)}$ (Theorem 1.6), as well as exponential lower bounds for this model, simply by applying the results for ROABPs. It is still not clear, however, how to handle the general case, since even read-2 oblivious ABPs are exponentially stronger than ROABPs (recall that [38] give an exponential separation between a sum of two ROABPs and ROABPs, and we separate 2-pass varying-order ABPs from sums of ROABPs).

1.4.3 PIT for read-k oblivious ABPs

Let us focus, for the time being, on the simplest instance of the more general problem, by considering a 2-pass varying-order ABP computing a non-zero polynomial f. That is, an ABP of width w that, without loss of generality, reads the variables in the order $x_1, x_2, \ldots, x_n, x_{\pi(1)}, x_{\pi(2)}, \ldots, x_{\pi(n)}$, for some permutation π. As we mentioned, we cannot possibly hope to simulate any such branching program by a small ROABP. We do, however, find a large subset of the variables S, such that if we fix all the other variables arbitrarily (or, equivalently, think of f as a polynomial in the variables of S over the field of rational functions $\mathbb{F}(\overline{S})$), the resulting polynomial has a small ROABP.

By the well-known Erdős–Szekeres Theorem [20], any sequence of distinct integers of length n contains either a monotonically increasing subsequence of length $\sqrt{n}$, or a monotonically decreasing subsequence of the same length. Applied to the sequence $x_{\pi(1)}, x_{\pi(2)}, \ldots, x_{\pi(n)}$ (with the natural order $x_1 < x_2 < \cdots < x_n$) we get a monotone subsequence of variables, which we might as well — for the sake of this exposition — assume to be monotonically increasing (the case of a decreasing sequence is, somewhat counter-intuitively, even simpler). Let $S = \{y_1, \ldots, y_{\sqrt{n}}\}$ be the set of $\sqrt{n}$ elements that appear in this monotone subsequence. Having fixed all the variables in $\overline{S}$, we are left, by the monotonicity property, with a branching program that reads the variables in the order $y_1, y_2, \ldots, y_{\sqrt{n}}, y_1, y_2, \ldots, y_{\sqrt{n}}$. Observe that this is exactly a 2-pass branching program! Hence, the previous arguments apply here, and if f is non-zero, we can efficiently find an assignment to the variables in S from $\mathbb{F}$ that keeps the polynomial non-zero. Having reached this point, we can "resurrect" the variables in $\overline{S}$, but note that we are left with only $n - \sqrt{n}$ variables. These are again computed by a 2-pass varying-order ABP, so me may apply the same argument repeatedly. After $O(\sqrt{n})$

iterations we are guaranteed to find an assignment to all the variables on which f evaluates to a non-zero output.

At each stage, we construct a hitting set for width-poly(w) ROABPs, of size $(nwd)^{O(\log n)}$. Since we take a cartesian product over $O(\sqrt{n})$ sets, the total size of the hitting set will eventually be $(nwd)^{\tilde{O}(\sqrt{n})}$, as promised by Theorem 1.7.

Generalizing the argument above for k-pass varying-order ABPs is fairly straightforward, and is done using repeated applications of the Erdős–Szekeres Theorem to each of the k sequences in order to obtain a subsequence of a subset of the variables S which is monotone in every pass and has size only $n^{1/2^{k-1}}$, which accounts for most of the loss in the parameters.[2] The polynomial, restricted to variables in S, will be computed by a k-pass ABP.

In order to handle general read-k oblivious ABPs, we need more ideas. We observe that after repeatedly applying the Erdős–Szekeres Theorem to the subsequence of every "read", we do not get a k-pass ABP as before, but rather k monotone sequences that are intertwined together. We next show that by discarding more variables, but not too many, we get a structure that we call a "k-regularly interleaving sequence". This is a technical notion which is presented in full details in Section 5, but the main point is that this definition allows us to argue that the obtained read-k oblivious ABP has a (small) evaluation dimension and therefore it can be simulated by a not-too-large ROABP. Obtaining this k-regularly interleaving property is the main technical difficulty of the proof.

1.4.4 Lower bounds for read-k oblivious ABPs

The arguments above that give PIT algorithms already give lower bounds for read-k oblivious ABPs. We have shown that if f is computed by a 2-pass varying-order ABP of width w, then there exist a subset of $\sqrt{n}$ variables S such that f is computed by an ROABP of width w^4 over $\mathbb{F}(\overline{S})$. This implies that if we pick f so that every restriction to $\sqrt{n}$ variables has an exponential (in $\sqrt{n}$) lower bound for ROABPs, we would receive a subexponential lower bound for computing f in a 2-pass varying-order ABP. (These arguments, again, generalize to read-k oblivious ABPs.)

In order to get an exponential lower bound (Theorem 1.5), we observe that we do not need to bound the evaluation dimension for *every* prefix (namely, to show that a subset of the variables is computed by a small ROABP), but only to show that the evaluation dimension is small for *some* prefix. This is much easier to achieve since we do not need the order of the reads to be "nicely-behaved" with respect to *every* prefix, but just with respect to *a* prefix.

In other words, we invoke a simple averaging argument to show that if f is computed by a width-w read-k oblivious ABP, then there exist sets of variables S (of size at least $n/k^{O(k)}$) and T (of size at most $n/100$), so that whenever we fix the variables in T we get that $\text{evalDim}_{S,\overline{S}}(g) \leq w^{2k}$, where g is any restriction of f obtained by fixing the variables in T. We then construct an explicit polynomial whose evaluation dimension with respect to every set remains large, even after arbitrarily fixing a small set of the variables (see Theorem 4.2).

1.4.5 Separating 2-pass ABPs from sums of ROABPs

In order to prove the separation with a 2-pass varying-order ABPs and sum of c ROABPs (Theorem 1.8), we use a structural result proved by Gurjar et al. [29] that gives a way

[2] This lower bound on the length of a subsequence which is monotone in every pass is the best possible. This fact is attributed to de Bruijn (unpublished, see [45]), and the actual construction which shows that the lower bound is tight appears in [6].

to argue by induction on ROABPs. Given a polynomial f which is computed by a sum $h_1 + h_2 + \cdots h_c$ of ROABPs of width w, we would like to find a related polynomial f' that is computed by a sum of $c-1$ ROABPs of perhaps slightly larger width. Here, the evaluation dimension plays a role as well. The way to do this is to pick a non-trivial linear combination of $w+1$ partial evaluations of f that make h_1 zero, which is possible since h_1 has a small evaluation dimension with respect to prefixes of variables corresponding to the order in which the variables are read in h_1. One can then show that, having eliminated h_1, each of the other summands can still be computed by a ROABP of width $w(w+1)$.

We provide a simple polynomial computed by a 2-pass varying-order ROABP whose partial evaluations are complex enough in the sense that they contain many linear independent evaluations and also a "scaled-down" version of the original polynomial as a projection. It then follows by induction, using the above arguments, that this polynomial cannot be computed by a small sum of small ROABPs (see Lemma 3.3).

1.5 Organization

We start with some preliminaries and useful facts about the evaluation dimension in Section 2 that almost all the results in this paper rely on. In Section 3, we present the separation between the class of 2-pass varying order ABPs and sums of ROABPs. Following that, in Section 4, we present an exponential lower bound for the class of general read-k oblivious ABPs. Then in Section 5 we present the white-box PIT for read-k oblivious ABPs. Finally, we conclude with some open problems in Section 6.

The proofs of some of the results are omitted from this version, and can be found in the full version of the paper ([7]).

2 Preliminaries

2.1 Notation

For $n \in \mathbb{N}$, we denote by $[n]$ the set $\{1, 2, \ldots, n\}$. We commonly denote by $\mathbf{x}$ a set of n indeterminates $\{x_1, \ldots, x_n\}$, where the number of indeterminates n is understood from the context. As we often deal with prefixes of this set, we denote by $\mathbf{x}_{[i]}$ the set $\{x_1, \ldots, x_i\}$, and more generally, for any $S \subseteq [n]$, $\mathbf{x}_S$ denotes the set $\{x_i \mid i \in S\}$.

For a polynomial $f \in \mathbb{F}[\mathbf{x}]$, a set $S \subseteq [n]$ and vector $\mathbf{a} = (a_1, \ldots, a_{|S|}) \in \mathbb{F}^{|S|}$, we denote by $f|_{\mathbf{x}_S = \mathbf{a}}$ the restriction of f obtained by fixing the j-th element in S to a_j.

For a subset $S \subseteq \mathbf{x}$ of variables, we denote its complement by $\overline{S}$. For disjoint subsets $S, T \subseteq [n]$ we denote by $S \sqcup T$ their disjoint union.

In our PIT algorithm, we need to combine hitting sets for smaller sets of variables. Hence, for a partition of $[n]$, $S_1 \sqcup S_2 \sqcup \cdots \sqcup S_m = [n]$, and sets $\mathcal{H}_i \subseteq \mathbb{F}^{|S_i|}$, we denote by $\mathcal{H}_1^{S_1} \times \cdots \times \mathcal{H}_m^{S_m}$ the set of all vectors in $\mathbb{F}^n$ whose restriction to the S_i coordinates is an element of $\mathcal{H}_i$, that is

$$\mathcal{H}_1^{S_1} \times \cdots \times \mathcal{H}_m^{S_m} = \{v \in \mathbb{F}^n \mid \forall i \in [m],\ v|_{S_i} \in \mathcal{H}_i\}.$$

We will also use the following theorem that gives a construction of a hitting set for ROABPs.

▶ **Theorem 2.1** (Hitting Set for ROABPs, [2]). *There exists a hitting set $\mathcal{H}$ for the class of n-variate polynomials computed by width-w individual-degree-d ROABPs of size $(nwd)^{O(\log n)}$, in any variable order. $\mathcal{H}$ can be constructed in time* $\mathrm{poly}(|\mathcal{H}|)$.

2.2 ABPs and iterated matrix products

The computation of an ABP corresponds to iterated multiplication of matrices of polynomials. In the case of oblivious branching programs, the ABP computes an iterated matrix product of univariate matrices. We record this fact as a lemma, and refer to [21] for a proof and a detailed discussion on this subject.

▶ **Lemma 2.2.** *Suppose f is a polynomial computed by an oblivious ABP A of width w and length ℓ, that reads the variables in some order $x_{i_1}, x_{i_2}, \ldots, x_{i_\ell}$. Then f is the $(1,1)$ entry of a matrix of the form*

$$A_1(x_{i_1}) \cdot A_2(x_{i_2}) \cdots A_\ell(x_{i_\ell})$$

where for every $j \in [\ell]$, $A_j \in \mathbb{F}[x_{i_j}]^{w \times w}$ is a $w \times w$ matrix in which each entry is a univariate polynomial in x_{i_j}.

2.3 Evaluation dimension and ROABPs

We now define a complexity measure for polynomials that we will use frequently when analyzing read-k oblivious ABPs.

▶ **Definition 2.3** (Evaluation dimension)**.** Let $f \in \mathbb{F}[x_1, \ldots, x_n]$ be a polynomial, and $S = \{x_{i_1}, \ldots, x_{i_r}\}$ be subset of variables. We define $\mathrm{eval}_S(f)$ to be

$$\mathrm{eval}_S(f) = \mathrm{span}\left\{ f|_{\mathbf{x}_S = \mathbf{a}} \ : \ \mathbf{a} \in \mathbb{F}^r \right\} \subseteq \mathbb{F}[\overline{S}],$$

which is the space of polynomials spanned by all partial evaluations of the S variables in f.

If $\mathbf{x} = S \sqcup T \sqcup R$ we define the *evaluation dimension of f with respect to $S \sqcup T$ over* $\mathbb{F}(\mathbf{x}_R)$, which shall be denoted by $\mathrm{evalDim}_{S,T;R}(f)$, as the dimension of the space $\mathrm{eval}_S(f)$ when taken over the field of rational functions $\mathbb{F}(\mathbf{x}_R)$. That is, we first "move" the variables $\mathbf{x}_R$ into the field and treat them as constants, and then consider the dimension of $\mathrm{eval}_S(f)$ over $\mathbb{F}(\mathbf{x}_R)$.

In the special case where $R = \emptyset$, we shall just use the notation $\mathrm{evalDim}_{S,T}(f)$.

If $|\mathbb{F}| > \deg(f)$, then $\mathrm{evalDim}_{S,T}(f)$ is the rank of the *partial derivative matrix with respect to S,T*, as defined by Nisan [50]. The rows of the partial derivative matrix are indexed by monomials m_S in S and its columns are indexed by monomials m_T in T. The (m_S, m_T) entry is the coefficient of $m_S m_T$ in the polynomial f. Although these two perspectives are equivalent, the formulation via evaluations is sometimes easier to work with. The evaluation dimension measure is useful when arguing about ROABPs since it characterizes the width needed to compute a polynomial f using a ROABP.

▶ **Theorem 2.4** ([50], and see also [21])**.** *Let f be a polynomial on $\mathbf{x} = \{x_1, \ldots, x_n\}$ and suppose for every $i \in [n]$ we have $\mathrm{evalDim}_{\mathbf{x}_{[i]}, \overline{\mathbf{x}_{[i]}}}(f) \leq w$. Then, there is a ROABP of width w in the order $x_1, \ldots, x_n$ that computes f.*

Conversely, if $\mathrm{evalDim}_{\mathbf{x}_{[i]}, \overline{\mathbf{x}_{[i]}}}(f) = w$, then in any ROABP that computes f in the order $x_1, x_2, \ldots, x_n$, the width of the i-th layer must be at least w.

Let us give an example of a polynomial which has large evaluation dimension with respect to a specific subset. This example will be helpful not only because it is simple to argue about, but also because all of our constructions of hard polynomials later on will ultimately be based on a reduction to this case.

▶ **Lemma 2.5.** *Let $f(\mathbf{u},\mathbf{v},\mathbf{w})$ be a polynomial of the form*

$$f = \left(\prod_{i=1}^{t}(\ell_i(\mathbf{v}) + \ell'_i(\mathbf{u}))\right) \cdot g(\mathbf{u},\mathbf{w}),$$

where:
1. *For every $\mathbf{a} \in \mathbb{F}^{|\mathbf{u}|}$, it holds that $g|_{\mathbf{u}=\mathbf{a}} = g(\mathbf{a},\mathbf{w}) \not\equiv 0$.*
2. *$\{\ell_i\}_{i=1}^t$ is a set of linearly-independent linear functions , and so is $\{\ell'_i\}_{i=1}^t$.*
Then $\mathrm{evalDim}_{\mathbf{u},\mathbf{v}\sqcup\mathbf{w}}(f) \geq 2^t$.

The proof of Lemma 2.5 appears in the full version of the paper [7].

The following simple lemma is an illustration of using the evaluation dimension of a polynomial to obtain a small ROABP for that polynomial.

▶ **Lemma 2.6.** *Let $f \in \mathbb{F}[x_1,\ldots,x_n]$ be a polynomial computed by a k-pass ABP of width w, according to the order π. Then f can be computed by a width-w^{2k} read-once ABP in the order π.*

Proof. Let A be the k-pass ABP computing f. We may assume without loss of generality that the k passes of A read the variables in the order $x_1,\ldots,x_n$. Recall that for any $i \in [n]$, we denote $\mathbf{x}_{[i]} = \{x_1,\ldots,x_i\}$. By Theorem 2.4, it is enough to show that for any $i \in [n]$,

$$\mathrm{evalDim}_{\mathbf{x}_{[i]},\overline{\mathbf{x}_{[i]}}} \leq w^{2k}.$$

By the assumption on f and by Lemma 2.2, for every $i \in [n]$ and $j \in [k]$ there exists a matrix $M^{i,j} \in \mathbb{F}^{w\times w}$ such that the entries of $M^{i,j}$ are univariate polynomials in x_i and

$$f = \left(M^{1,1}(x_1)M^{2,1}(x_2)\cdots M^{n,1}(x_n)M^{1,2}(x_1)M^{2,2}(x_2)\cdots M^{n,k}(x_n)\right)_{1,1}.$$

Fix $i \in [n]$, and consider any assignment of the form $\mathbf{x}_{[i]} = \mathbf{a}$ for $\mathbf{a} = (a_1,\ldots,a_i) \in \mathbb{F}^i$. Having fixed $\mathbf{x}_{[i]}$, we get that for some k matrices $N_1(\mathbf{a}),\ldots,N_k(\mathbf{a})$, that depend on $\mathbf{a}$,

$$\begin{aligned} f|_{\mathbf{x}_{[i]}=\mathbf{a}} = \big(&N_1(\mathbf{a})\cdot M^{i+1,1}(x_{i+1})\cdots M^{n,1}(x_n)\cdot N_2(\mathbf{a})\cdot M^{i+1,2}(x_{i+1})M^{n,2}(x_n) \\ &\cdots N_k(\mathbf{a})\cdot M^{i+1,k}(x_{i+1})\cdots M^{n,k}(x_n)\big)_{1,1}. \end{aligned} \tag{1}$$

It follows that any polynomial $g(x_{i+1},\ldots,x_n) \in \mathrm{eval}_{\mathbf{x}_{[i]}}(f)$ is completely determined by $N_1,\ldots,N_k$ which have w^2 entries each. More precisely, let $\{B_1,\ldots,B_{w^2}\}$ be a basis for $\mathbb{F}^{w\times w}$. For each $j \in [k]$, we can write $N_j(\mathbf{a}) \in \mathbb{F}^{w\times w}$ in (1) as a linear combination of $\{B_1,\ldots,B_{w^2}\}$. Then, by expanding the matrix product in (1), we see that every polynomial of the form $f|_{\mathbf{x}_{[i]}=\mathbf{a}}$ (and as a consequence, every polynomial in $\mathrm{eval}_{\mathbf{x}_{[i]}}(f)$) is spanned by the w^{2k} polynomials of the form

$$\begin{aligned} \big(&B_{\sigma_1}\cdot M^{i+1,1}(x_{i+1})\cdots M^{n,1}(x_n)\cdot B_{\sigma_2}\cdot M^{i+1,2}(x_{i+1})M^{n,2}(x_n)\cdots \\ &B_{\sigma_k}\cdot M^{i+1,k}(x_{i+1})\cdots M^{n,k}(x_n)\big)_{1,1} \end{aligned}$$

for $\sigma_1,\ldots,\sigma_k \in [w^2]$, which implies that $\mathrm{evalDim}_{\mathbf{x}_{[i]},\overline{\mathbf{x}_{[i]}}}(f) \leq w^{2k}$.

By Theorem 2.4, the claim follows. ◀

In fact, the proof of Lemma 2.6 permits a slight generalization of the lemma, by requiring weaker assumptions on the ABP, which is captured by the following definition.

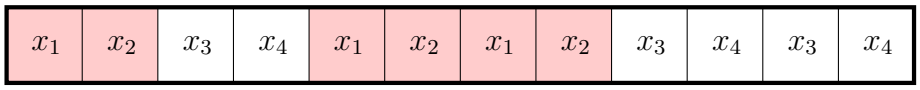


Figure 1 An ABP that reads the variables in this (left-to-right) order is a read-3 ABP that has the 2-gap property with respect to $\{x_1, x_2\}$.

▶ **Definition 2.7.** Let A be an ABP that computes a polynomial $f \in \mathbb{F}[x_1, \ldots, x_n]$. We say that A has the *k-gap property* with respect to $\{x_1, \ldots, x_i\}$, if there exist k matrices $M_1, \ldots, M_k \in \mathbb{F}^{w \times w}[x_{i+1}, \ldots, x_n]$ such that for every $\mathbf{a} \in \mathbb{F}^i$, there exists k matrices $N_1(\mathbf{a}), \ldots, N_k(\mathbf{a}) \in \mathbb{F}^{w \times w}$ such that

$$f|_{\mathbf{x}_i = \mathbf{a}} = \big(N_1(\mathbf{a}) \cdot M_1(x_{i+1}, \ldots, x_n) \cdot N_2(\mathbf{a}) \cdot M_2(x_{i+1}, \ldots, x_n) \\ \cdots N_k(\mathbf{a}) \cdot M_k(x_{i+1}, \ldots, x_n)\big)_{1,1}. \tag{2}$$

A is said to simply have the k-gap property if it has this property with respect to $\mathbf{x}_{[i]}$, for every $i \in [n]$.

Figure 1 provides a pictorial explanation for the choice of this terminology.

Using the exact same arguments as in the proof of Lemma 2.6, we obtain the following lemma.

▶ **Lemma 2.8.** *Let $f \in \mathbb{F}[x_1, \ldots, x_n]$ be a polynomial computed by an ABP of width w with the k-gap property. Then f can be computed by a width-w^{2k} read-once ABP.*

3 Separating 2-pass ABPs from sums of ROABPs

Recall that every *sum* of c ROABPs can be realized by an oblivious read-c ABP. In order to motivate our study of read-k oblivious ABP, we begin by showing a polynomial that can be computed by a constant-width, 2-pass varying-order ABP, and yet cannot be computed by a small sum of polynomial-size ROABPs. Thus, even a weak, but non-trivial, form of read-k oblivious ABPs, for $k = 2$, is already stronger than sums of ROABPs.

Suppose $\mathbf{x} = \{x_{1,1}, \ldots, x_{n,n}\}$ is a set of n^2 variables. It is useful to think of $\mathbf{x}$ as an $n \times n$ matrix X such that $x_{i,j}$ appears in the (i,j)-th entry. For every $m \in [n]$, define

$$\mathrm{rowSum}_m = \sum_j x_{m,j} \quad \text{and} \quad \mathrm{colSum}_m = \sum_i x_{i,m}.$$

Let

$$P_n(\mathbf{x}) = \left(\prod_{i=1}^{n} \mathrm{rowSum}_i\right) \cdot \left(\prod_{j=1}^{n} \mathrm{colSum}_j\right). \tag{3}$$

Observe that for all i, j, rowSum_i and colSum_j can be computed by width-2 ROABPs. Moreover, both $\prod_{i=1}^{n} \mathrm{rowSum}_i$ and $\prod_{j=1}^{n} \mathrm{colSum}_j$ can be as well. Indeed, their product P_n is computed by a 2-pass varying-order ABP.

▶ **Theorem 3.1.** *Let $P_n(x_{1,1}, \ldots, x_{n,n})$ be the n^2-variate polynomial defined in (3). For every $c > 0$, any sum of c ROABPs that computes it must have width $\exp(\sqrt{n}/2^c)$.*

The proof, which can be found in the full version of this paper, exploits the structure of a sum of few ROABPs that Gurjar, Korwar, Saxena and Thierauf [29] used for constructing hitting sets. The following lemma is essentially present implicitly in their result.

▶ **Lemma 3.2** ([29]). *Let $f = h_1 + \cdots + h_c$ where each h_i is computed by a width-w ROABP in possibly different orders. Then, for every $0 < t < n$, there exists a subset S of t variables such that for every set of $w+1$ partial assignments $\mathbf{a}_1, \ldots, \mathbf{a}_{w+1} \in \mathbb{F}^t$, there is some non-trivial linear combination of $\{f|_{\mathbf{x}_S = \mathbf{a}_i}\}_{i=1}^{t}$ that is computable by a sum of $c-1$ ROABPs of width $w(w+1)$ in possibly different orders. That is, there exists $\alpha_1, \ldots, \alpha_{w+1} \in \mathbb{F}$, not all zero, such that*

$$\sum_{i=1}^{w+1} \alpha_i \cdot f|_{\mathbf{x}_S = \mathbf{a}_i} = f'_1 + \cdots + f'_{c-1}$$

where each f'_i is a ROABP of width at most $w(w+1)$.

The following lemma shows that the polynomial P_n defined in (3) has many linearly independent partial evaluations.

▶ **Lemma 3.3.** *Let S be any subset of $\mathbf{x} = \{x_{1,1}, \ldots, x_{n,n}\}$ of size $t < n$. Then there exists $r \geq 2^{\sqrt{t}}$ partial evaluations $\mathbf{a}_1, \cdots, \mathbf{a}_r \in \{0,1\}^t \subseteq \mathbb{F}^t$ such that the polynomials $\{P_n|_{\mathbf{x}_S = \mathbf{a}_1}, \ldots, P_n|_{\mathbf{x}_S = \mathbf{a}_r}\}$ are linearly independent.*

Furthermore, for any $g \in \operatorname{span}\{P_n|_{\mathbf{x}_S = \mathbf{a}_i} \mid i \in [r]\}$, there is a set $\mathbf{y} \subseteq \mathbf{x} \setminus S$ of $(n-t-1)^2$ variables, such that $P_{n-t-1}(\mathbf{y})$ can be obtained as a projection of g: namely, for $\mathbf{z} = \mathbf{x} \setminus (\mathbf{y} \cup S)$ we can find $\mathbf{a} \in \mathbb{F}^{n-|S|-|\mathbf{y}|}$ such that $g|_{\mathbf{z}=\mathbf{a}} = P_{n-t-1}(\mathbf{y})$.

4 Lower bounds for read-k oblivious ABPs

In this section we show an explicit polynomial that has a polynomial-size depth-3 multilinear circuit and yet cannot be computed efficiently by a read-k oblivious ABP.

4.1 An explicit polynomial with large evaluation dimension

Raz and Yehudayoff [55] constructed an explicit multilinear polynomial $f(\mathbf{x})$ with evaluation dimension as high as possible with respect to any partition $S, \overline{S}$. Our requirements are slightly different, as we would need some "robustness" property, namely, we would want to argue that the evaluation dimension of the polynomial remains high even when we fix a small constant fraction (say, $n/10$) of the variables. Later, in Theorem 4.3, we show why this property implies hardness for read-k oblivious ABPs.

Our construction is inspired by a recent similar construction of Kayal, Nair and Saha [38].

Consider the complete bipartite graph $K_{n,n}$ with n vertices on each side. We shall label the left vertices as $x_1, \ldots, x_n$ and the right vertices as $y_1, \cdots, y_n$. We can write $K_{n,n}$ as a union of n edge-disjoint perfect matchings $M_1 \cup \cdots \cup M_n$, where for every $i \in [n]$, M_i contains all edges of the form $(x_j, y_{j+i \bmod n})$ for $j \in [n]$. Define the polynomial Q_n as

$$\begin{aligned} Q_n(x_1, \ldots, x_n, y_1, \ldots, y_n, z_1, \ldots, z_n) &= \sum_{i=1}^{n} z_i \prod_{(j,k) \in M_i} (x_j + y_k) \\ &= \sum_{i=1}^{n} z_i \prod_{j=1}^{n} (x_j + y_{j+i \bmod n}). \end{aligned} \tag{4}$$

By its definition, it is clear that Q_n is computed by a depth-3 polynomial-size circuit. We now show that even if we fix a small fraction of the variables in $\mathbf{x} \cup \mathbf{y}$, Q_n retains a large evaluation dimension with respect to any partition of the variables we have not fixed.

▶ **Lemma 4.1.** *Let S, T be two disjoint subsets of $\mathbf{x} \cup \mathbf{y}$ such that $|S \sqcup T| \geq 0.9 \cdot 2n$. Let $R = \mathbf{x} \cup \mathbf{y} \setminus (S \cup T)$. Then,*

$$\text{evalDim}_{S,T;R}(Q_n) \quad \geq \quad \exp(\Omega(\min(|S|, |T|))).$$

Proof. Assume without loss of generality that $|S| \leq |T|$, and that $S_L := S \cap \mathbf{x}$ satisfies $|S_L| \geq |S|/2$. Since $(S \cup T) \cap \mathbf{y} \geq 0.8n$, $|T \cap \mathbf{y}| \geq (0.8n - |S|/2) \geq 0.3n$. Thus, there are $\Omega(n \cdot |S|)$ edges between S and T in $K_{n,n}$. By averaging, some matching M_i must include at least $\Omega(|S|)$ of these edges. Consider the polynomial $f_i = \prod_{(j,k)\in M_i}(x_j + y_k)$. As $\Omega(|S|)$ of the edges in M_i go between S and T, we can write

$$f_i = \prod_{m=1}^{t} (u_m + v_m) \cdot g(\mathbf{w}),$$

where for every $m \in [t]$ we have that $u_m \in S$, $v_m \in T$, and $t = \Omega(|S|)$ (we have "pushed" to g all the factors that correspond to edges in the matching which do not go between S and T).

By Lemma 2.5, $\text{evalDim}_{S,T;R}(f_i) \geq 2^{\Omega(|S|)}$. Since f_i is a projection of Q_n (under the setting $z_i = 1$ and $z_j = 0$ for all $j \neq i$) it follows that $\text{evalDim}_{S,T;R}(Q_n) \geq \text{evalDim}_{S,T;R}(f_i) \geq \exp(\Omega(|S|))$. ◀

4.2 Upper bound on evaluation dimension for read-k oblivious ABPs

In this section we show that if f is computed by a read-k oblivious ABP of width w, then we can fix a "small" subset of variables such that the remaining variables can be partitioned into two carefully chosen "large" subsets, under which the evaluation dimension is at most w^{2k}. We then apply this result to the polynomial Q_n (from (4)) to show that if Q_n is computed by a width-w read-k oblivious ABP, then $w \geq \exp(n/k^{O(k)})$.

▶ **Theorem 4.2.** *Let $f \in \mathbb{F}[x_1, \ldots, x_n]$ be a polynomial computed by a width-w read-k oblivious ABP. Then, there exist three disjoint subsets $U \sqcup V \sqcup W = [n]$, such that*

1. $|U|, |V| \geq n/k^{O(k)}$,
2. $|W| \leq n/10$, *and*
3. $\text{evalDim}_{U,V;W}(f) \leq w^{2k}$.

Proof. Consider an ABP A that computes f. Divide the kn layers into r equal-sized contiguous blocks of kn/r layers (where r shall be set shortly). For each variable, consider the (at most) k blocks that its k reads fall in (if the number of such blocks is strictly smaller than k, we can fill up to k blocks arbitrarily). By a simple averaging, there must exist k blocks $B_1, \ldots, B_k$ that contain all k reads of a set U of at least $n/\binom{r}{k}$ variables. Let W be the set of variables in $B_1 \cup B_2 \cup \cdots \cup B_k$ that are not in U, and V be the set of all remaining variables. As each block is of size kn/r, we have that $|W| \leq k^2 n/r$, which is at most $n/10$ if we set $r = 10k^2$. Observe that $|V| \geq n - k^2 n/r \geq 9n/10$. Let us ignore the variables in W by considering the ABP over the field $\mathbb{F}(\mathbf{x}_W)$.

We now claim that $\text{evalDim}_{U,V;W}(f) \leq w^{2k}$. Having moved the variables in W to the field, each of the r blocks is either entirely contained in U or entirely contained in V. Therefore, since the reads comprise of at most k alternating blocks of variables in U and V, the resulting branching program has the k-gap property with respect to U. It follows immediately from Lemma 2.8 that $\text{evalDim}_{U,V;W}(f)$ is at most w^{2k}. ◀

We now show that Q_n (defined in (4)) is hard to compute for read-k oblivious ABPs.

▶ **Theorem 4.3.** *Let A be a width-w, read-k oblivious ABP computing the polynomial Q_n (defined in (4)). Then $w \geq \exp(n/k^{O(k)})$.*

Proof. First observe that we can eliminate the $\mathbf{z}$ variables by considering the ABP over the field $\mathbb{F}(\mathbf{z})$ so that it now computes a polynomial in the variables $\mathbf{x} \cup \mathbf{y}$.

By Theorem 4.2, there exists a partition $U \sqcup V \sqcup W$ of $\mathbf{x} \cup \mathbf{y}$ with the prescribed sizes as in the statement of the theorem, such that $\text{evalDim}_{U,V;W}(Q_n) \leq w^{2k}$.

Since $|W| \leq 2n/10$, Lemma 4.1 implies that $\text{evalDim}_{U,V;W}(f) = \exp(\Omega(\min(|U|, |V|)))$.

Using the fact that $\min(|U|, |V|) \geq n/k^{O(k)}$, we get that $w^{2k} \geq \exp(n/k^{O(k)})$, which implies $w \geq \exp(n/k^{O(k)})$ as well. ◀

5 Identity tests for read-k oblivious ABPs

5.1 Identity tests for k-pass ABPs

In this section we give PIT algorithms for the class of read-k oblivious ABPs. First, observe that Lemma 2.6 immediately implies a black-box algorithm for the subclass of k-pass ABPs, as those can be simulated efficiently by a ROABP.

▶ **Corollary 5.1.** *There is a hitting set of size $(ndw)^{O(k \log n)}$ for the class of n-variate k-pass ABPs of width w and degree d.*

Proof. Follows directly from Lemma 2.6 and the $(ndw')^{O(\log n)}$-sized hitting set for width w' read-once ABPs from Theorem 2.1. ◀

We now turn to general read-k oblivious ABPs. We again omit the proofs, which can be found in [7].

5.2 From read-k to per-read-monotone and regularly-interleaving sequences

In this section we show that given any read-k oblivious ABP over $\mathbf{x} = \{x_1, \ldots, x_n\}$ computing a polynomial f, we can find a "large" subset of variables $\mathbf{y} \subseteq \mathbf{x}$ such that f has a "small" ROABP when we think of f as a polynomial in the $\mathbf{y}$ variables over the field $\mathbb{F}(\overline{\mathbf{y}})$. This process, in fact, involves only finding the correct subset $\mathbf{y}$ (without rewiring any part of the ABP). Therefore, in order to avoid technical overhead it is useful to think in terms sequences over abstract sets of elements, which correspond to the order in which the ABP reads the variables, and not in terms of variables in branching programs.

Let X be a set, and let $n = |X|$. Let $S \in X^m$ be an sequence of elements from X. We say S is *read-k* if each element $x \in X$ occurs k times in S (in this case we also have $m = nk$). As mentioned in Remark 1.3, we will restrict ourselves to considering sequences that are read-k for some k. For $i \in [k]$, we denote by $S^{(i)}$ the subsequence of S which consists of the i-th occurrences of elements in X. That is, $S^{(i)}$ is a permutation of the elements of X, according to the order in which they appear in S for the i-th time. Similarly, for $i \neq j \in [k]$, we use the notation $S^{(i,j)}$ for the subsequence of S which consists of the i-th and j-th occurrences of elements in X.

For a subset $X' \subseteq X$, let $S|_{X'}$ denote the restriction of S to the set X' that is the result of dropping all elements of $X \setminus X'$ from S. Thus, $S|_{X'} \in X^{m'}$ for $m' = |X'|k$.

In order to save on excessive notation and multiple indexing, we will assume without loss of generality that $S^{(1)} = (x_1, \ldots, x_n)$, that is, that the variables in $\mathbf{x}$ are already labeled

according to the order of their first occurrence. This can be ensured by renaming variables, if necessary.

Next we define a special subclass of read-k sequences which we work with throughout this section.

▶ **Definition 5.2.** Let $S \in X^{nk}$ be a read-k sequence. We say S is *per-read-monotone* if for every $i \in [k]$, $S^{(i)}$ is monotone (that is, the variables all appear in either increasing or decreasing order).

The following well-known theorem asserts that any long enough sequence contains a large monotone subsequence:

▶ **Theorem 5.3** (Erdős–Szekeres Theorem, [20, 4])**.** *Let S be a sequence of distinct integers of length at least m^2+1. Then, there exists a monotonically increasing subsequence of S of length $m+1$, or a monotonically decreasing subsequence of S of length $m+1$.*

As an immediate corollary of Theorem 5.3, we get the following lemma:

▶ **Lemma 5.4.** *Let S be a read-2 sequence over $X = \{x_1, \ldots, x_n\}$. Then, there exists a subset $X' \subseteq X$ with $|X'| \geq \sqrt{n-1}+1 \geq \sqrt{n}$ such that the subsequence $S' = S|_{X'}$ is per-read-monotone.*

We can generalize Lemma 5.4 to read-k sequences, at the cost of settling for a weaker lower bound of only $n^{1/2^{k-1}}$ on the length of the subsequence:

▶ **Lemma 5.5.** *Let S be a read-k sequence over $X = \{x_1, \ldots, x_n\}$. Then, there exists a subset $X' \subseteq X$ with $|X'| \geq n^{1/2^{k-1}}$ such that the subsequence $S' = S|_{X'}$ is per-read-monotone.*

We now show how to prune per-read-monotone read-2 sequences even further, trading a constant fraction of their size for stronger structural properties. We begin by stating the property we look for.

▶ **Definition 5.6.** Let S be a read-2 sequence over a set of elements X. We say S is *2-regularly-interleaving* if there exists a partition of X to *blocks* $\{X_i\}_{i\in[t]}$ such that for every $i \in [t]$:

- For every $c \in \{1,2\}$, all the c-th occurrences of the block X_i appear consecutively in S.
- The interval containing the second occurrences of the block X_i immediately follows the interval containing the first occurrences of X_i.

A read-k sequence S is said to be *k-regularly-interleaving* if for any $i \neq j \in [k]$, the subsequence $S^{(i,j)}$ is 2-regularly-interleaving. That is, S is k-regularly-interleaving if restricted to any two reads it is 2-regularly-interleaving.

To get a better intuitive sense of the definition, the reader may consult Figure 2.

An example of a read-k sequence that is *k-regularly interleaving* is 12121234345656345 6.

The following lemma is used to simplify some of the later arguments. It shows that in a read-k per-read-monotone sequence, the monotonically increasing subsequences cannot intersect with monotonically decreasing subsequences.

▶ **Lemma 5.7.** *Let S be a read-k, per-read-monotone sequence over $X = \{x_1, \ldots, x_n\}$. Suppose $S^{(1)}$ is monotonically increasing. Then we can write S as a concatenation $S = (T_1, T_2, \ldots, T_t)$, such that:*

1. *for every $j \in [t]$, T_j is a read-k_j sequence for $k_j \leq k$.*
2. *for every $i \in [k]$ there exists $j \in [t]$ so that $S^{(i)}$ is contained in T_j.*

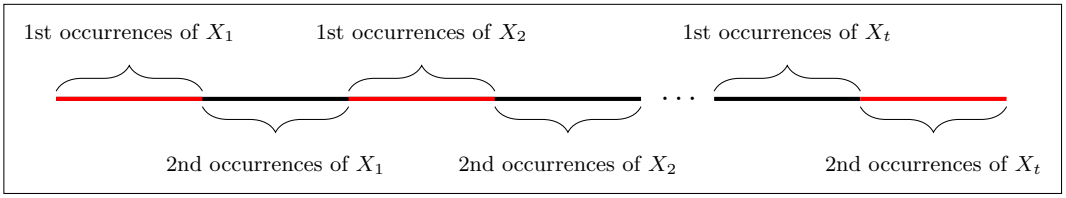


Figure 2 A 2-regularly-interleaving sequence.

3. *for every odd $j \in [t]$, all the subsequences $S^{(i)}$ that appear in T_j are monotonically increasing, and for any even j, all are monotonically decreasing.*
4. *for every $j \in [t-1]$, the last element that appears in T_j equals the first element appearing in T_{j+1}, and this element can be either x_n (if T_j contains monotonically increasing subsequences and T_{j+1} contains monotonically decreasing subsequences) or x_1 (in the opposite case).*

In other words, we can partition S into t disjoint contiguous subsequences, such that every $S^{(i)}$ is completely contained in exactly one subsequence, and in every subsequence, either all reads are increasing or all reads are decreasing, with the pattern alternating.

The following lemma shows that given a 2-read per-read-monotone sequence, we can find a large subsequence which is also 2-regularly interleaving.

▶ **Lemma 5.8.** *Let S be a read-2 per-read-monotone sequence over $X = \{x_1, \ldots, x_s\}$. Then there is a subset $X' \subseteq X$ with $|X'| \geq s/3$ such that the sequence $S' = S|_{X'}$ is per-read-monotone and 2-regularly-interleaving.*

Viewed as an algorithmic process, the proof of Lemma 5.8 is a procedure that, given a per-read-monotone sequence S over X, decides which elements of X should be erased in order to be left with a 2-regular-interleaving sequence $S' = S|_{X'}$. It can also be noted that both properties of being per-read-monotone and being 2-regularly interleaving are downward-closed, in the sense that if we now take a subset $X'' \subseteq X'$ and look at $S'' = S'|_{X''}$, it will maintain both properties. Hence, if we are given a read-k per-read-monotone sequence S, by repeatedly applying the algorithmic process of Lemma 5.8 separately on each subsequence $S^{(i,j)}$ for $i \neq j \in [k]$ (maintaining a constant fraction of the elements on each application), we get the following corollary:

▶ **Corollary 5.9.** *Let S be a read-k per-read-monotone sequence over $X = \{x_1, \ldots, x_s\}$. Then there is a subset $X' \subseteq X$ with $|X'| \geq s/3^{k^2}$ such that the sequence $S' = S|_{X'}$ is per-read-monotone and k-regularly-interleaving.*

5.3 ROABPs for regularly interleaving sequences

In this section we show that if a polynomial f is computed by a small-width read-k oblivious ABP A such that the sequence S of the reads in A is per-read-monotone and k-regularly-interleaving, then f can in fact also be computed by a small-width ROABP A' (in the same order as $S^{(1)}$). We show this by proving that A has the k-gap property with respect to that order, and then applying Lemma 2.8.

▶ **Lemma 5.10.** *Let $f \in \mathbb{F}[x_1, \ldots, x_n]$ be computed by a read-k oblivious ABP A of width w, and let S be the sequence of variables read by A. Suppose further that S is per-read-monotone (with respect to the order $x_1 < x_2 < \cdots < x_n$) and k-regularly-interleaving. Then A has the k-gap property.*

It now immediately follows that any read-k oblivious ABP the reads the variables in a per-read-monotone and k-regularly-interleaving fashion can be simulated by a small ROABP. We record this fact in the following corollary.

▶ **Corollary 5.11.** *Let $f \in \mathbb{F}[x_1, \ldots, x_n]$ be computed by a read-k oblivious ABP A of width w, and let S be the sequence of variables read by A. Suppose further that S is per-read-monotone (with respect to the order $x_1 < x_2 < \cdots < x_n$) and k-regularly-interleaving. Then for any $i \in [n]$, $\text{evalDim}_{\mathbf{x}_{[i]}, \overline{\mathbf{x}_{[i]}}}(f) \leq w^{2k}$. In particular, f is computed by a ROABP of width at most w^{2k} in the variable order $x_1, x_2, \ldots, x_n$.*

5.4 Identity testing for read-k oblivious ABPs

In this section we give our white-box identity testing algorithm for read-k oblivious ABPs. Before giving the proof, let us first give an overview of the algorithm for the slightly simpler read-2 case.

Given a read-2 oblivious ABP A with read sequence S which computes a polynomial $f \in \mathbb{F}[x_1, \ldots, x_n]$, Lemma 5.8 shows how to find a read-2 subsequence on a set $\mathbf{y} = \{y_1, \ldots, y_{\sqrt{n}}\}$ of roughly $\sqrt{n}$ variables, such that when we think of f as a polynomial in the $\mathbf{y}$ variables over the field $\mathbb{F}(\overline{\mathbf{y}})$, it has a small ROABP. We can then use a hitting set for ROABPs in order to find an assignment (from $\mathbb{F}$) to the $\mathbf{y}$ variables that keeps the polynomial non-zero. Having done that, we are left with a non-zero polynomial over a smaller set of $n - \sqrt{n}$ variables, which is again computed by a read-2 oblivious ABP, so we may repeat this process. After at most $O(\sqrt{n})$ iterations we find an assignment for all the variables that keeps the polynomial non-zero. We note that a very similar "hybrid argument" that uses a hitting set for ROABPs appears both in [2] and [53].

The argument for read-k is identical, apart from the loss in the parameters incurred by Corollary 5.9.

▶ **Theorem 5.12.** *There is a white-box polynomial identity test for read-k oblivious ABPs of width w and degree d on n variables that runs in time $\text{poly}(n, w, d)^{n^{1-1/2^{k-1}} \exp(k^2) \text{polylog}(n)}$. Furthermore, given only the order in which the variables are read, we can construct a hitting set for such ABPs that read their variables in this order, of size $\text{poly}(n, w, d)^{n^{1-1/2^{k-1}} \exp(k^2) \text{polylog}(n)}$.*

Our PIT algorithm is presented in Algorithm 1.

Algorithm 1 : PIT for read-k oblivious ABPs

Input: a read-k oblivious ABP A computing a polynomial $f \in \mathbb{F}[x_1, \ldots, x_n]$.

1: $\mathbf{x} = \{x_1, \ldots, x_n\}$, $i = 1$
2: **while** $\mathbf{x} \neq \emptyset$ **do**
3: Pick a subset $\mathbf{y}_i \subseteq \mathbf{x}$ of size at least $|\mathbf{x}|^{1-1/2^{k-1}}/3^{k^2}$, such that the subsequence that reads only the $\mathbf{y}_i$ variables is per-read-monotone and k-regularly-interleaving (such a subset exists by Lemma 5.5 and Corollary 5.9).
4: Construct a set $\mathcal{H}_i \subseteq \mathbb{F}^{|\mathbf{y}_i|}$ of size $(nw^{2k}d)^{O(\log n)}$ that hits ROABPs of width w^{2k} in the $\mathbf{y}_i$ variables, using Theorem 2.1.
5: $\mathbf{x} \leftarrow \mathbf{x} \setminus \mathbf{y}_i$, $i \leftarrow i + 1$
6: **end while**
7: **return** the set $\mathcal{H} = \mathcal{H}_1^{\mathbf{y}_1} \times \cdots \times \mathcal{H}_t^{\mathbf{y}_t}$ (where t is the number of iterations of the loop).

6 Conclusions and Open Problems

In this work, we have obtained the first non-trivial lower bounds and identity testing algorithms for read-k oblivious ABPs. We briefly mention some directions that we find worth pursuing for future research.

The most natural open problem we pose is designing an identity testing algorithm for read-k oblivious ABPs with better running time than the algorithm we presented in this paper. Since for ROABPs (the $k = 1$ case) there exist a white-box polynomial time and black-box quasipolynomial-time algorithms, it seems reasonable to hope that the deterioration in the parameters would not be as sharp when $k > 1$ (the flip side of this argument, however, is the relative lack of progress in the analogous question in the boolean domain).

Another open problem is obtaining a complete black-box test for read-k oblivious ABPs, in any variable order (that is, without knowing the order in which the variable appear). As we mentioned, for ROABPs there exist a black-box hitting set that works for any variable order [2], whose size is essentially the same as that of the hitting set that was obtained earlier for the known order case [24]. In our construction, we need to know the order so that we can pick the per-read-monotone and k-regularly-interleaving sequences to which we assign the hitting sets for ROABPs, and simply "guessing" those sets would require exponential time. Still, given the progress in obtaining hitting sets in any order for ROABPs, it might be the case that such a construction could follow from our strategy, even using known techniques.

Finally, we turn back to boolean complexity, and ask whether our ideas and techniques can be adapted to attack the problem of constructing pseudorandom generators for read-k oblivious *boolean* branching program with sublinear seed length.

References

1 Manindra Agrawal. Proving Lower Bounds Via Pseudo-random Generators. In *Proceedings of the 25th International Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2005)*, pages 92–105, 2005. doi:10.1007/11590156_6.

2 Manindra Agrawal, Rohit Gurjar, Arpita Korwar, and Nitin Saxena. Hitting-sets for ROABP and sum of set-multilinear circuits. *SIAM Journal of Computing*, 44(3):669–697, 2015. doi:10.1137/140975103.

3 Manindra Agrawal and V. Vinay. Arithmetic circuits: A chasm at depth four. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2008)*, pages 67–75, 2008. doi:10.1109/FOCS.2008.32.

4 Martin Aigner and Günter M. Ziegler. *Proofs from THE BOOK*. Springer, 2004. doi:10.1007/978-3-662-44205-0.

5 Miklós Ajtai. A non-linear time lower bound for boolean branching programs. *Theory of Computing*, 1(1):149–176, 2005. Preliminary version in the *40th Annual IEEE Symposium on Foundations of Computer Science (FOCS 1999)*. doi:10.4086/toc.2005.v001a008.

6 Noga Alon, Zoltán Füredi, and Meir Katchalski. Separating pairs of points by standard boxes. *European Journal of Combinatorics*, 6(3):205–210, 1985. doi:10.1016/S0195-6698(85)80028-7.

7 Matthew Anderson, Michael A. Forbes, Ramprasad Saptharishi, Amir Shpilka, and Ben Lee Volk. Identity testing and lower bounds for read-k oblivious algebraic branching programs. *Electronic Colloquium on Computational Complexity (ECCC)*, 22:184, 2015. URL: http://eccc.hpi-web.de/report/2015/184.

8 Matthew Anderson, Dieter van Melkebeek, and Ilya Volkovich. Derandomizing Polynomial Identity Testing for Multilinear Constant-Read Formulae. In *Proceedings of the 26th Annual IEEE Conference on Computational Complexity (CCC 2011)*, pages 273–282, 2011.

9 Vikraman Arvind and Sathasivam Raja. Some lower bound results for set-multilinear arithmetic computations. *Electronic Colloquium on Computational Complexity (ECCC)*, 22:176, 2015. URL: http://eccc.hpi-web.de/report/2015/176/.

10 László Babai, Péter Hajnal, Endre Szemerédi, and György Turán. A lower bound for read-once-only branching programs. *J. Comput. Syst. Sci.*, 35(2):153–162, 1987. doi:10.1016/0022-0000(87)90010-9.

11 Paul Beame, Michael E. Saks, Xiaodong Sun, and Erik Vee. Time-space trade-off lower bounds for randomized computation of decision problems. *J. ACM*, 50(2):154–195, 2003. doi:10.1145/636865.636867.

12 Andrej Bogdanov, Zeev Dvir, Elad Verbin, and Amir Yehudayoff. Pseudorandomness for width-2 branching programs. *Theory of Computing*, 9:283–293, 2013. doi:10.4086/toc.2013.v009a007.

13 Andrej Bogdanov, Periklis A. Papakonstantinou, and Andrew Wan. Pseudorandomness for read-once formulas. In *Proceedings of the 52nd Annual IEEE Symposium on Foundations of Computer Science (FOCS 2011)*, pages 240–246, 2011. doi:10.1109/FOCS.2011.57.

14 Allan Borodin, Alexander A. Razborov, and Roman Smolensky. On lower bounds for read-k-times branching programs. *Computational Complexity*, 3:1–18, 1993. doi:10.1007/BF01200404.

15 Mark Braverman, Anup Rao, Ran Raz, and Amir Yehudayoff. Pseudorandom generators for regular branching programs. *SIAM J. Comput.*, 43(3):973–986, 2014. doi:10.1137/120875673.

16 Anindya De. Pseudorandomness for permutation and regular branching programs. In *Proceedings of the 26th Annual IEEE Conference on Computational Complexity (CCC 2011)*, pages 221–231, 2011. doi:10.1109/CCC.2011.23.

17 Richard A. DeMillo and Richard J. Lipton. A Probabilistic Remark on Algebraic Program Testing. *Information Processing Letters*, 7(4):193–195, 1978. doi:10.1016/0020-0190(78)90067-4.

18 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. Preliminary version in the *37th Annual ACM Symposium on Theory of Computing (STOC 2005)*. doi:10.1137/05063605X.

19 Zeev Dvir, Amir Shpilka, and Amir Yehudayoff. Hardness-randomness tradeoffs for bounded depth arithmetic circuits. *SIAM J. Comput.*, 39(4):1279–1293, 2009. doi:10.1137/080735850.

20 Paul Erdős and George Szekeres. A combinatorial problem in geometry. *Compositio Math.*, 2:463–470, 1935. doi:10.1007/978-0-8176-4842-8_3.

21 Michael Forbes. *Polynomial Identity Testing of Read-Once Oblivious Algebraic Branching Programs*. PhD thesis, Massachusetts Institute of Technology, 2014. URL: http://hdl.handle.net/1721.1/89843.

22 Michael A. Forbes, Ramprasad Saptharishi, and Amir Shpilka. Hitting sets for multilinear read-once algebraic branching programs, in any order. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC 2014)*, pages 867–875, 2014. doi:10.1145/2591796.2591816.

23 Michael A. Forbes and Amir Shpilka. Explicit noether normalization for simultaneous conjugation via polynomial identity testing. In *Proceedings of the 17th International Workshop on Randomization and Computation (RANDOM 2013)*, volume 8096 of *Lecture Notes in Computer Science*, pages 527–542. Springer, 2013. doi:10.1007/978-3-642-40328-6_37.

24 Michael A. Forbes and Amir Shpilka. Quasipolynomial-time identity testing of non-commutative and read-once oblivious algebraic branching programs. In *Proceedings of the*

54th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2013), pages 243–252, 2013. Full version at arXiv:1209.2408. doi:10.1109/FOCS.2013.34.

25 Hervé Fournier, Nutan Limaye, Guillaume Malod, and Srikanth Srinivasan. Lower bounds for depth 4 formulas computing iterated matrix multiplication. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC 2014)*, pages 128–135, 2014. doi:10.1145/2591796.2591824.

26 Parikshit Gopalan, Raghu Meka, Omer Reingold, Luca Trevisan, and Salil P. Vadhan. Better pseudorandom generators from milder pseudorandom restrictions. In *Proceedings of the 53rd Annual IEEE Symposium on Foundations of Computer Science (FOCS 2012)*, pages 120–129. IEEE Computer Society, 2012. doi:10.1109/FOCS.2012.77.

27 Ankit Gupta, Pritish Kamath, Neeraj Kayal, and Ramprasad Saptharishi. Arithmetic Circuits: A Chasm at Depth Three. In *Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2013)*, pages 578–587, 2013. doi:10.1109/FOCS.2013.68.

28 Ankit Gupta, Pritish Kamath, Neeraj Kayal, and Ramprasad Saptharishi. Approaching the chasm at depth four. *Journal of the ACM*, 61(6):33:1–33:16, 2014. Preliminary version in the *28th Annual IEEE Conference on Computational Complexity (CCC 2013)*. doi:10.1145/2629541.

29 Rohit Gurjar, Arpita Korwar, Nitin Saxena, and Thomas Thierauf. Deterministic identity testing for sum of read-once oblivious arithmetic branching programs. In *Proceedings of the 30th Annual Computational Complexity Conference (CCC 2015)*, pages 323–346, 2015. doi:10.4230/LIPIcs.CCC.2015.323.

30 Joos Heintz and Claus-Peter Schnorr. Testing Polynomials which Are Easy to Compute (Extended Abstract). In *Proceedings of the 12th Annual ACM Symposium on Theory of Computing (STOC 1980)*, pages 262–272, 1980. doi:10.1145/800141.804674.

31 Russell Impagliazzo, Raghu Meka, and David Zuckerman. Pseudorandomness from shrinkage. In *Proceedings of the 53rd Annual IEEE Symposium on Foundations of Computer Science (FOCS 2012)*, pages 111–119, 2012. doi:10.1109/FOCS.2012.78.

32 Russell Impagliazzo, Noam Nisan, and Avi Wigderson. Pseudorandomness for network algorithms. In *Proceedings of the 26th Annual ACM Symposium on Theory of Computing (STOC 1994)*, pages 356–364, 1994. doi:10.1145/195058.195190.

33 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. URL: http://eccc.hpi-web.de/report/2010/084.

34 Valentine Kabanets and Russell Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Computational Complexity*, 13(1-2):1–46, 2004. Preliminary version in the *35th Annual ACM Symposium on Theory of Computing (STOC 2003)*. doi:10.1007/s00037-004-0182-6.

35 Zohar Shay Karnin, Partha Mukhopadhyay, Amir Shpilka, and Ilya Volkovich. Deterministic identity testing of depth-4 multilinear circuits with bounded top fan-in. *SIAM Journal of Computing*, 42(6):2114–2131, 2013. Preliminary version in the *42nd Annual ACM Symposium on Theory of Computing (STOC 2010)*. doi:10.1137/110824516.

36 Richard M. Karp, Eli Upfal, and Avi Wigderson. Constructing a perfect matching is in random NC. *Combinatorica*, 6(1):35–48, 1986. Preliminary version in the *17th Annual ACM Symposium on Theory of Computing (STOC 1985)*. doi:10.1007/BF02579407.

37 Neeraj Kayal, Nutan Limaye, Chandan Saha, and Srikanth Srinivasan. An Exponential Lower Bound for Homogeneous Depth Four Arithmetic Circuits. In *Proceedings of the 55th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2014)*, 2014. doi:10.1109/FOCS.2014.15.

38 Neeraj Kayal, Vineet Nair, and Chandan Saha. Separation between read-once oblivious algebraic branching programs (roabps) and multilinear depth three circuits. *Electronic Colloquium on Computational Complexity (ECCC)*, 22:154, 2015. URL: http://eccc.hpi-web.de/report/2015/154/.

39 Neeraj Kayal, Chandan Saha, and Ramprasad Saptharishi. A super-polynomial lower bound for regular arithmetic formulas. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC 2014)*, pages 146–153, 2014. doi:10.1145/2591796.2591847.

40 Neeraj Kayal and Shubhangi Saraf. Blackbox polynomial identity testing for depth-3 circuits. In *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2009)*, 2009. doi:10.1109/FOCS.2009.67.

41 Neeraj Kayal and Nitin Saxena. Polynomial identity testing for depth 3 circuits. *Computational Complexity*, 16(2):115–138, 2007. doi:10.1007/s00037-007-0226-9.

42 Pascal Koiran. Arithmetic circuits: The chasm at depth four gets wider. *Theoretical Computer Science*, 448:56–65, 2012. doi:10.1016/j.tcs.2012.03.041.

43 Swastik Kopparty, Shubhangi Saraf, and Amir Shpilka. Equivalence of polynomial identity testing and polynomial factorization. *Computational Complexity*, 24(2):295–331, 2015. Preliminary version in the *29th Annual IEEE Conference on Computational Complexity (CCC 2014)*. doi:10.1007/s00037-015-0102-y.

44 Michal Koucký, Prajakta Nimbhorkar, and Pavel Pudlák. Pseudorandom generators for group products: extended abstract. In *Proceedings of the 43rd Annual ACM Symposium on Theory of Computing (STOC 2011)*, pages 263–272, 2011. doi:10.1145/1993636.1993672.

45 Joseph B Kruskal. Monotonic subsequences. *Proceedings of the American Mathematical Society*, 4(2):264–274, 1953. doi:10.1090/S0002-9939-1953-0053256-2.

46 Mrinal Kumar and Shubhangi Saraf. The limits of depth reduction for arithmetic formulas: it's all about the top fan-in. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC 2014)*, pages 136–145, 2014. doi:10.1145/2591796.2591827.

47 Mrinal Kumar and Shubhangi Saraf. On the power of homogeneous depth 4 arithmetic circuits. In *Proceedings of the 55th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2014)*, 2014. doi:10.1109/FOCS.2014.46.

48 Ketan Mulmuley. Geometric complexity theory V: equivalence between blackbox derandomization of polynomial identity testing and derandomization of noether's normalization lemma. In *Proceedings of the 53rd Annual IEEE Symposium on Foundations of Computer Science (FOCS 2012)*, pages 629–638, 2012. doi:10.1109/FOCS.2012.15.

49 Ketan Mulmuley, Umesh V. Vazirani, and Vijay V. Vazirani. Matching is as easy as matrix inversion. *Combinatorica*, 7(1):105–113, 1987. Preliminary version in the *19th Annual ACM Symposium on Theory of Computing (STOC 1987)*. doi:10.1007/BF02579206.

50 Noam Nisan. Lower bounds for non-commutative computation. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing (STOC 1991)*, pages 410–418, 1991. doi:10.1145/103418.103462.

51 Noam Nisan. Pseudorandom generators for space-bounded computation. *Combinatorica*, 12(4):449–461, 1992. doi:10.1007/BF01305237.

52 E.A. Okolnishnikova. Lower bounds on the complexity of realization of characteristic functions of binary codes by branching programs. *Metody Diskretnogo Analiza*, 51:61–83, 1991. URL: http://www.thi.informatik.uni-frankfurt.de/~jukna/boolean/Russians/Okolnishnikova-1991.pdf.

53 Rafael Oliveira, Amir Shpilka, and Ben Lee Volk. Subexponential size hitting sets for bounded depth multilinear formulas. In *Proceedings of the 30th Annual Computational Complexity Conference (CCC 2015)*, pages 304–322, 2015. doi:10.4230/LIPIcs.CCC.2015.304.

54 Ran Raz and Amir Shpilka. Deterministic polynomial identity testing in non-commutative models. *Computational Complexity*, 14(1):1–19, 2005. Preliminary version in the *19th Annual IEEE Conference on Computational Complexity (CCC 2004)*. doi:10.1007/s00037-005-0188-8.

55 Ran Raz and Amir Yehudayoff. Lower bounds and separations for constant depth multilinear circuits. *Computational Complexity*, 18(2):171–207, 2009. Preliminary version in the *23rd Annual IEEE Conference on Computational Complexity (CCC 2008)*. doi:10.1007/s00037-009-0270-8.

56 Omer Reingold, Thomas Steinke, and Salil P. Vadhan. Pseudorandomness for regular branching programs via fourier analysis. In *Proceedings of the 17th International Workshop on Randomization and Computation (RANDOM 2013)*, pages 655–670, 2013. doi:10.1007/978-3-642-40328-6_45.

57 Shubhangi Saraf and Ilya Volkovich. Black-box identity testing of depth-4 multilinear circuits. In *Proceedings of the 43rd Annual ACM Symposium on Theory of Computing (STOC 2011)*, pages 421–430, 2011. doi:10.1145/1993636.1993693.

58 Nitin Saxena and C. Seshadhri. Blackbox identity testing for bounded top-fanin depth-3 circuits: The field doesn't matter. *SIAM J. Comput.*, 41(5):1285–1298, 2012. Preliminary version in the *43rd Annual ACM Symposium on Theory of Computing (STOC 2011)*. doi:10.1137/10848232.

59 Jacob T. Schwartz. Fast Probabilistic Algorithms for Verification of Polynomial Identities. *Journal of the ACM*, 27(4):701–717, 1980. doi:10.1145/322217.322225.

60 Amir Shpilka and Ilya Volkovich. On the relation between polynomial identity testing and finding variable disjoint factors. In *Proceedings of the 37th International Colloquium on Automata, Languages and Programming (ICALP 2010)*, pages 408–419, 2010. doi:10.1007/978-3-642-14165-2_35.

61 Amir Shpilka and Ilya Volkovich. Read-once polynomial identity testing. *Computational Complexity*, 24(3):477–532, 2015. Preliminary version in the *40th Annual ACM Symposium on Theory of Computing (STOC 2008)*. doi:10.1007/s00037-015-0105-8.

62 Amir Shpilka and Amir Yehudayoff. Arithmetic circuits: A survey of recent results and open questions. *Foundations and Trends in Theoretical Computer Science*, 5:207–388, March 2010. doi:10.1561/0400000039.

63 Thomas Steinke. Pseudorandomness for permutation branching programs without the group theory. *Electronic Colloquium on Computational Complexity (ECCC)*, 19:83, 2012. URL: http://eccc.hpi-web.de/report/2012/083.

64 Thomas Steinke, Salil P. Vadhan, and Andrew Wan. Pseudorandomness and fourier growth bounds for width-3 branching programs. In *Proceedings of the 18th International Workshop on Randomization and Computation (RANDOM 2014)*, pages 885–899, 2014. doi:10.4230/LIPIcs.APPROX-RANDOM.2014.885.

65 Sébastien Tavenas. Improved bounds for reduction to depth 4 and depth 3. *Inf. Comput.*, 240:2–11, 2015. Preliminary version in the *38th Internationl Symposium on the Mathematical Foundations of Computer Science (MFCS 2013)*. doi:10.1016/j.ic.2014.09.004.

66 Jayram S. Thathachar. On separating the read-k-times branching program hierarchy. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing (STOC 1998)*, pages 653–662, 1998. doi:10.1145/276698.276881.

67 Leslie G. Valiant, Sven Skyum, S. Berkowitz, and Charles Rackoff. Fast Parallel Computation of Polynomials Using Few Processors. *SIAM Journal of Computing*, 12(4):641–644, 1983. Preliminary version in the *6th Internationl Symposium on the Mathematical Foundations of Computer Science (MFCS 1981)*. doi:10.1137/0212043.

68 Ingo Wegener. On the complexity of branching programs and decision trees for clique functions. *J. ACM*, 35(2):461–471, 1988. doi:10.1145/42282.46161.

69 Stanislav Zák. An exponential lower bound for one-time-only branching programs. In *Proceedings of the 9th Internationl Symposium on the Mathematical Foundations of Computer Science (MFCS 1984)*, volume 176 of *Lecture Notes in Computer Science*, pages 562–566. Springer, 1984. doi:10.1007/BFb0030340.

70 Richard Zippel. Probabilistic algorithms for sparse polynomials. In *Symbolic and Algebraic Computation, EUROSAM'79, An International Symposiumon Symbolic and Algebraic Computation*, volume 72 of *Lecture Notes in Computer Science*, pages 216–226. Springer, 1979. doi:10.1007/3-540-09519-5_73.

Reconstruction of Real Depth-3 Circuits with Top Fan-In 2

Gaurav Sinha

Department of Mathematics, California Institute of Technology, Pasadena, USA
gsinha@caltech.edu

Abstract

Reconstruction of arithmetic circuits has been heavily studied in the past few years and has connections to proving lower bounds and deterministic identity testing. In this paper we present a polynomial time randomized algorithm for reconstructing $\Sigma\Pi\Sigma(2)$ circuits over $\mathbb{F}$ ($char(\mathbb{F}) = 0$), i.e. depth-3 circuits with fan-in 2 at the top addition gate and having coefficients from a field of characteristic 0.

The algorithm needs only a blackbox query access to the polynomial $f \in \mathbb{F}[x_1, \ldots, x_n]$ of degree d, computable by a $\Sigma\Pi\Sigma(2)$ circuit C. In addition, we assume that the "*simple rank*" of this polynomial (essential number of variables after removing the gcd of the two multiplication gates) is bigger than a fixed constant. Our algorithm runs in time $poly(n, d)$ and returns an equivalent $\Sigma\Pi\Sigma(2)$ circuit (with high probability).

The problem of reconstructing $\Sigma\Pi\Sigma(2)$ circuits over finite fields was first proposed by Shpilka [24]. The generalization to $\Sigma\Pi\Sigma(k)$ circuits, $k = O(1)$ (over finite fields) was addressed by Karnin and Shpilka in [15]. The techniques in these previous involve iterating over all objects of certain kinds over the ambient field and thus the running time depends on the size of the field $\mathbb{F}$. Their reconstruction algorithm uses lower bounds on the lengths of Linear Locally Decodable Codes with 2 queries. In our settings, such ideas immediately pose a problem and we need new ideas to handle the case of the characteristic 0 field $\mathbb{F}$.

Our main techniques are based on the use of Quantitative Sylvester Gallai Theorems from the work of Barak et al. [3] to find a small collection of "*nice*" subspaces to project onto. The heart of our paper lies in subtle applications of the Quantitative Sylvester Gallai theorems to prove why projections w.r.t. the "*nice*" subspaces can be "glued". We also use Brill's Equations from [8] to construct a small set of candidate linear forms (containing linear forms from both gates). Another important technique which comes very handy is the polynomial time randomized algorithm for factoring multivariate polynomials given by Kaltofen [14].

1998 ACM Subject Classification G.1.1 Interpolation, I.4.5 Reconstruction

Keywords and phrases Reconstruction, $\Sigma\Pi\Sigma(2)$, Sylvester-Gallai, Brill's Equations

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.31

1 Introduction

The last few years have seen significant progress towards interesting problems dealing with arithmetic circuits. Some of these problems include Deterministic Polynomial Identity Testing, Reconstruction of Circuits and recently Lower Bounds for Arithmetic Circuits. There has also been work connecting these three different aspects. In this paper we will primarily be concerned with the reconstruction problem. Even though it's connections to Identity Testing and Lower Bounds are very exciting, the problem in itself has drawn a lot of attention because of elegant techniques and connections to learning. The strongest version of the problem requires that for any $f \in \mathbb{F}[x_1, \ldots, x_n]$ with blackbox access given one wants to

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 31; pp. 31:1–31:53

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

construct (roughly) most succint representation i.e. the smallest possible arithmetic circuit computing the polynomial. This general problem appears to be very hard. Most of the work done has dealt with some special type of polynomials i.e. the ones which exhibit constant depth circuits with alternating addition and multiplication gates. Our result adds to this by looking at polynomials computed by circuits of this type (alternating addition/multiplication gates but of depth 3). Our circuits will have variables at the leaves, operations $(+, \times)$ at the gates and scalars at the edges. We also assume that the top gate has only two children and the "*simple rank*" of this polynomial (essential number of variables after removing the gcd of the two multiplication gates) is bigger than a constant. The bottom most layer has addition gates and so computes linear forms, the middle layer then multiplies these linear forms together and the top layer adds two such products. Later in Remark 1.1 we discuss that we may assume the linear forms computed at bottom level to be homogeneous and the in-degree of all gates at middle level to be the same (= degree of f). Therefore these circuits compute polynomials with the following form:

$$f(x_1, \ldots, x_n) = G(x_1, \ldots, x_n)(T_0(x_1, \ldots, x_n) + T_1(x_1, \ldots, x_n))$$

where $T_i(x_1, \ldots, x_n) = \prod_{j=1}^{M} l_{ij}$ and $G(x_1, \ldots, x_n) = \prod_{j=1}^{d-M} G_j$ with the l_{ij}'s and G_j's being linear forms for $i \in \{0, 1\}$. Also assume $gcd(T_0, T_1) = 1$. Our condition about the essential number of variables (after removing gcd from the multiplication gates) is called "*simple rank*" of the polynomial and is defined as dimension of the space

$$sp\{l_{ij} : i \in \{0, 1\}, j \in \{1, \ldots, M\}\}$$

When the underlying field $\mathbb{F}$ is of characteristic 0 ($\mathbb{Q}, \mathbb{R}$ or $\mathbb{C}$ for simplicity), we give an efficient randomized algorithm for reconstructing the circuit representation of such polynomials. Formally our main theorem reads:

▶ **Theorem 1.1** ($\Sigma\Pi\Sigma_{\mathbb{F}}(2)$ Reconstruction Theorem). *Let $f = G(T_0 + T_1) \in \mathbb{F}[x_1, \ldots, x_n]$ be any degree d, $n-$ variate polynomial (to which we have blackbox access) which can be computed by a depth 3 circuit with top fan-in 2 (i.e. a $\Sigma\Pi\Sigma(2)$ circuit) i.e. G, T_i being products of affine forms. Assume $gcd(T_0, T_1) = 1$ and $span\{l : l \mid T_0T_1\}$ is bigger than $s + 1$ (a fixed constant defined below). We give a randomized algorithm which runs in time $poly(n, d)$ and computes the cicuit for f with high probability.*

▶ **Definition 1.2.** We fix s to be any constant $> \max(C_{2k-1} + k, c_{\mathbb{F}}(4))$ where:

1. $c_{\mathbb{F}}(l) = 3l^2$ is the rank lower bound (see Theorem 1.7) that guarantees non-zeroness of any simple, minimal, $\Sigma\Pi\Sigma(l)$ circuit with rank $> c_{\mathbb{F}}(l)$.
2. $k = c_{\mathbb{F}}(3) + 2$.
3. δ is some fixed number in $(0, \frac{7-\sqrt{37}}{6})$.
4. $C_k = \frac{C^k}{\delta}$ the constant that appears in Theorem B.4.

From our discussion before the theorem about Remark 1.1, we can assume in the above theorem that the polynomial and all linear forms involved are homogeneous.

As per our knowledge this is the first algorithm that efficiently reconstructs such circuits (over the char 0 fields). Over finite fields, the same problem has been considered by [24] and our method takes inspiration from their work. They also generalized this finite field version to circuits with arbitrary (but constant) top fan-in in [15]. However we need many new tools and techniques as their methods don't generalize at a lot of crucial steps. For example:

- They iterate through linear forms in a finite field which we unfortunately cannot do.
- They use lower bounds for Locally Decodable Codes given in [7] which again does not work in our setup.

We resolve these issues by

- Constructing candidate linear forms by solving simultaneous polynomial equations obtained from Brill's Equations (Chapter 4, [8]).
- Using quantitative versions of the Sylvester Gallai Theorems given in [3] and [6]. This new method enables us to construct *nice* subspaces, take projections onto them and glue the projections back to recover the cicuit representation.

1.1 Previous Work and Connections

Efficient Reconstruction algorithms are known for some concrete class of circuits. We list some here:

- Depth-2 $\Sigma\Pi$ circuits (sparse polynomials) in [20]
- Read-once arithmetic formulas in [25]
- Non-commutative ABP's [2]
- $\Sigma\Pi\Sigma(2)$ circuits over finite fields in [24], extended to $\Sigma\Pi\Sigma(k)$ circuits (over finite fields) with $k = O(1)$ in [15].
- Random Multilinear Formular in [11]
- Depth 4 ($\Sigma\Pi\Sigma\Pi$) multilinear circuits with top fan-in 2 in [10]
- Random Arithmetic Formulas in [12]

All of the above work introduced new ideas and techniques and have been greatly appreciated.

It's straightforward to observe that a polynomial time deterministic reconstruction algorithm for a circuit class C also implies a polynomial time Deterministic Identity Testing algorithm for the same class. From the works [1] and [13] it has been established that blackbox Identity Testing for certain circuit classes imply superpolynomial circuit lower bounds for an explicit polynomial. Hence the general problem of deterministic reconstruction cannot be easier than proving superpolynomial lower bounds. So one might first try and relax the requirements and demand a randomized algorithm. Another motivation to consider the probabilistic version comes from Learning Theory. A fundamental question called the *exact learning problem using membership queries* asks the following: *Given oracle access to a Boolean function, compute a small description for it.* This problem has attracted a lot of attention in the last few decades. For e.g. in [18][9] and [17] a negative result stating that a class of boolean circuits containing the trapdoor functions or pseudo-random functions has no efficient learning algorithms. Among positive works [23], [4], [19] show that when f has a small circuit (inside some restricted class) exact learning from membership queries is possible. Our problem is a close cousin as we are looking for exact learning algorithms for algebraic functions. Because of this connection with learning theory it makes sense to also allow randomized algorithms for reconstruction.

1.2 Depth-3 Arithmetic Circuits

We will use the definitions from [16]. Let C be an arithmetic circuit with coefficients in the field $\mathbb{F}$. We say C is a $\Sigma\Pi\Sigma(k)$ circuit if it computes an expression of the form:

$$C(\bar{x}) = \sum_{i\in[k]} \prod_{j\in[d]} l_{i,j}(\bar{x}) \,.$$

$l_{i,j}(\bar{x})$ are linear forms of the type $l_{i,j}(\bar{x}) = \sum_{s\in[n]} a_s x_s$ where $(a_1,\ldots,a_n) \in \mathbb{F}^n$ and $(x_1,\ldots,x_n)$ is an $n-$ tuple of indeterminates. For convenience we denote the multiplication gates in C as

$$T_i = \prod_{j\in[d]} l_{i,j}(\bar{x})\,.$$

k is the top fanin of our circuit C and d is the fanin of each multiplication gate T_i. With these definitions we will say that our circuit is of type $\Sigma\Pi\Sigma_{\mathbb{F}}(k,d,n)$. When most parameters are understood we will just call it a $\Sigma\Pi\Sigma(k)$ circuit.

▶ **Remark 1.1.** *Note that we are cosidering homogeneous circuits. There are two basic assumptions:*
1. *$l_{i,j}$'s have no constant term i.e. they are linear forms.*
2. *Fanin of each T_i is equal to d.*
If these are not satisfied we can homogenize our circuit by considering $Z^d(C(\frac{X_1}{Z},\ldots,\frac{X_n}{Z}))$. Now both the conditions will be taken care of by reconstructing this new homogenized circuit. We need a rank condition on our polynomial which remains essentially unchanged even after this substitution.

▶ **Definition 1.3** (Minimal Circuit)**.** We say that the circuit C is minimal if no strict non empty subsets of the $\Pi\Sigma$ polynomials $\{T_1,\ldots,T_k\}$ sums to zero.

▶ **Definition 1.4** (Simple Circuit and Simplification)**.** A circuit C is called Simple if the gcd of the $\Pi\Sigma$ polynomials $gcd(T_1,\ldots,T_k)$ is equal to 1 (i.e. is a unit). The simplification of a $\Sigma\Pi\Sigma(k)$ circuit C denoted as $Sim(C)$ is the $\Sigma\Pi\Sigma(k)$ circuit obtained by dividing each term by the gcd of all terms i.e.

$$Sim(C) \stackrel{def}{=} \sum_{i\in[k]} \frac{T_i}{gcd(T_1,\ldots,T_k)}\,.$$

▶ **Definition 1.5** (Rank of a Circuit)**.** Identifying each linear form $l(\bar{x}) = \sum_{s\in[n]} a_s x_s$ with the vector $(a_1,\ldots,a_n) \in \mathbb{F}^n$, we define the rank of C to be the dimension of the vector space spanned by the set $\{l_{i,j} | i \in [k], j \in [d]\}$.

▶ **Definition 1.6** (Simple Rank of a Circuit)**.** For a $\Sigma\Pi\Sigma(k)$ circuit C we define the *Simple Rank* of C as the rank of the circuit $Sim(C)$.

Before we go further into the paper and explain our algorithm we state some results about uniqueness of these circuits. In a nutshell for a $\Sigma\Pi\Sigma_{\mathbb{F}}(2,d,n)$ circuit C, if one assumes that the *Simple rank* of C is bigger than a constant ($c_{\mathbb{F}}(4)$: defined later) then the circuit is essentially unique.

1.3 Uniqueness of Representation

Shpilka et al. showed the uniqueness of circuit representation in [24] using rank bounds for Polynomial Identity Testing. The bound they used were from the work of Dvir et al. in [7]. It essentialy states that the rank of a simple, minimal $\Sigma\Pi\Sigma(k)$ circuit ($d \geq 2, k \geq 3$) which computes the identically zero polynomial is $\leq 2^{O(k^2)} \log^{k-2} d$. For circuits over char 0 fields improved rank bounds were given by Kayal et al. in [16].

In a series of following work the rank bounds for identically zero $\Sigma\Pi\Sigma(k)$ circuits got further improved. The best known bounds over char 0 fields were given by Saxena et al. in [22]. We rewrite Theorem 1.5 in [22] here for completion.

▶ **Theorem 1.7** (Theorem 1.5 in [22]). *Let C be a $\Sigma\Pi\Sigma(k,d,n)$ circuit over field $\mathbb{F}$ that is simple, minimal and zero. Then, $rk(C) < 3k^2$.*

Let $c_{\mathbb{F}}(k) = 3k^2$. This gives us the following version of Corollary 7, Section 2.1 in [24].

▶ **Theorem 1.8** ([24]). *Let $f(\bar{x}) \in \mathbb{F}[x]$ be a polynomial which exhibits a $\Sigma\Pi\Sigma(2)$ circuit*

$$C = G(A+B)\,.$$

$A = \prod_{j\in[M]} A_j, B = \prod_{j\in[M]} B_j, G = \prod_{i\in[d-M]} G_i$, *where $A_i, B_j, G_k \in Lin_{\mathbb{F}}[\bar{x}]$. $gcd(A,B) = 1$, and $Sim(C) = A+B$ has rank $\geq c_{\mathbb{F}}(4)+1$ then the representation is unique. That is if:*

$$f = G(A+B) = \tilde{G}(\tilde{A}+\tilde{B})$$

where $A, B, \tilde{A}, \tilde{B}$ are $\Pi\Sigma$ polynomials over $\mathbb{F}$ and $gcd(\tilde{A},\tilde{B}) = 1$ then we have $G = \tilde{G}$ and $(A,B) = (\tilde{A},\tilde{B})$ or $(\tilde{B},\tilde{A})$ (upto scalar multiplication).

Proof. Let $g = gcd(G,\tilde{G})$ and let $G = gG_1, \tilde{G} = g\tilde{G_1}$. Then $gcd(G_1,\tilde{G_1}) = 1$ and we get

$$G_1A + G_1B - \tilde{G_1}\tilde{A} - \tilde{G_1}\tilde{B} = 0$$

This is a simple $\Sigma\Pi\Sigma(4)$ circuit with *rank* bigger than $c_{\mathbb{F}}(4)+1$ and is identically 0 so it must be not minimal. Considering the various cases one can easily prove the required equality. ◀

1.4 Notation

$[n]$ denotes the set $\{1,2,\ldots,n\}$. Throughout the paper we will work over the field $\mathbb{F}$. Let V be a finite dimensional $\mathbb{F}$ vector space and $S \subset V$, $sp(S)$ will denote the linear span of elements of S. $dim(S)$ is the dimension of the subspace $sp(S)$. If $S = \{s_1,\ldots,s_k\} \subset V$ is a set of linearly independent vectors then $fl(S)$ denotes the affine subspace generated by points in S (also called a $(k-1)-flat$ or just $flat$ when dimension is understood). In particular:

$$fl(S) = \{\sum_{i=1}^{k} \lambda_i s_i : \lambda_i \in \mathbb{F}, \sum_{i=1}^{k} \lambda_i = 1\}\,.$$

Let $W \subset V$ be a subspace, then we can extend basis and get another subspace W' (called the complement of W) such that $W \oplus W' = V$. Note that the complement need not be unique. Corresponding to each such decomposition of V we may define orthogonal projections $\pi_W, \pi_{W'}$ onto W, W' respectively. Let $v = w + w' \in V, w \in W, w' \in W'$:

$$\pi_W(v) = w, \pi_{W'}(v) = w'\,.$$

$(\bar{x})$ will be used for the tuple $(x_1,\ldots,x_n)$.

$$Lin_{\mathbb{F}}[\bar{x}] = \{a_1x_1 + \ldots + a_nx_n : a_i \in \mathbb{F}\} \subset \mathbb{F}[\bar{x}]$$

is the vector space of all linear forms over the variables $(x_1,\ldots,x_n)$. For a linear form $l \in Lin_{\mathbb{F}}[\bar{x}]$ and a polynomial $f \in \mathbb{F}[x]$ we write $l \mid f$ if l divides f and $l \nmid f$ if it does not. We say $l^d \mid\mid f$ if $l^d \mid f$ but $l^{d+1} \nmid f$.

$$\Pi\Sigma^d_{\mathbb{F}}[\bar{x}] = \{l_1(\bar{x})\ldots l_d(\bar{x}) : l_i \in Lin_{\mathbb{F}}[\bar{x}]\} \subset \mathbb{F}[\bar{x}]$$

is the set of degree d homogeneous polynomials which can be written as product of linear forms. This collection for all possible d is called the set

$$\Pi\Sigma_{\mathbb{F}}[\bar{x}] = \bigcup_{d\in\mathbb{N}} \Pi\Sigma^d_{\mathbb{F}}[\bar{x}]$$

also called $\Pi\Sigma$ polynomials for convenience. Let $f(\bar{x}) \in \mathbb{F}[x]$ then $Lin(f) \in \Pi\Sigma_{\mathbb{F}}[\bar{x}]$ denotes the product of all linear factors of $f(\bar{x})$. Let $\mathcal{L}(f)$ denote the set of all linear factors of f. For any set of polynomials $S \subset \mathbb{C}[\bar{x}]$, we denote by $\mathbb{V}(S)$, the set of all complex simultaneous solutions of polynomials in S (this set is called the variety of S), i.e.

$$\mathbb{V}(S) = \{a \in \mathbb{C} : \text{ for all } f \in S, f(a) = 0\}.$$

Let $\mathcal{B} = \{b_1, \ldots, b_n\}$ be an ordered basis for $V = Lin_{\mathbb{F}}[\bar{x}]$. We define maps $\phi_{\mathcal{B}} : V \setminus \{0\} \to V$ as

$$\phi_{\mathcal{B}}(a_1b_1 + \ldots + a_nb_n) = \frac{1}{a_k}(a_1b_1 + \ldots + a_nb_n)$$

where k is such that $a_i = 0$ for all $i < k$ and $a_k \neq 0$.

A non-zero linear form l is called normal with respect to $\mathcal{B}$ if $l \in \Phi_{\mathcal{B}}(V)$ i.e. the first non-zero coefficient is 1. A polynomial $P \in \Pi\Sigma_{\mathbb{F}}[\bar{x}]$ is normal w.r.t. $\mathcal{B}$ if it is a product of normal linear forms. For two polynomials $P_1, P_2 \in \Pi\Sigma_{\mathbb{F}}[\bar{x}]$ we define:

$$gcd_{\mathcal{B}}(P_1, P_2) = P \in \Pi\Sigma_{\mathbb{F}}[\bar{x}], P \text{ normal w.r.t. } \mathcal{B} \text{ such that } P \mid P_1, P \mid P_2$$

When a basis is not mentioned we assume that the above definitions are with respect to the standard basis.

We can represent any linear form in $Lin_{\mathbb{F}}[\bar{x}]$ as a point in the vector space $\mathbb{F}^n$ and vice versa. To be precise we define the cannonical map $\Gamma : Lin_{\mathbb{F}}[\bar{x}] \to \mathbb{F}^n$ as

$$\Gamma(a_1x_1 + \ldots + a_nx_n) = (a_1, \ldots, a_n).$$

Γ is a linear isomorphism of vector spaces $Lin_{\mathbb{F}}[\bar{x}]$ and $\mathbb{F}^n$. Because of this isomorphism we will interchange between points and linear forms whenever we can. We choose to represent the linear form $a(\bar{x}) = a_1x_1 + \ldots + a_nx_n$ as the point $a = (a_1, \ldots, a_n)$.

LI will be the abbreviation for Linearly Independent and **LD** will be the abbreviation for Linearly Dependent.

▶ **Definition 1.9** (Standard Linear Form)**.** A non zero vector v is called *standard* with respect to basis $\mathcal{B} = \{b_1, \ldots, b_n\}$ if the coefficient of b_1 in v is 1. When a basis is not mentioned we assume we're talking about the standard basis. (Equivalently for linear forms the coefficient of x_1 is 1). A $\Pi\Sigma$ polynomial will be called *standard* if it is a product of standard linear forms.

We close this section with a lemma telling us when can we replace the span of some vectors with the affine span or flat. We've used this several times in the paper.

▶ **Lemma 1.10.** *Let $l, l_1, \ldots, l_t \in Lin_{\mathbb{F}}[\bar{x}]$ be* standard *linear forms w.r.t. some basis $\mathcal{B} = \{b_1, \ldots, b_n\}$ such that $l \in sp(\{l_1, \ldots, l_t\})$ then*

$$l \in fl(\{l_1, \ldots, l_t\}).$$

Proof. Since $l \in sp(\{l_1, \ldots, l_t\})$, we know that $l = \sum_{i \in [t]} \alpha_i l_i$ for some scalars $\alpha_i \in \mathbb{F}$. All linear forms are *standard* w.r.t. $\mathcal{B} \Rightarrow$ comparing the coefficients of b_1 we get that $\sum_{i \in [t]} \alpha_i = 1$ and therefore $l \in fl(\{l_1, \ldots, l_t\})$. ◀

Let $T \subset \mathbb{F}^n$, By a scaling of T we mean a set where all vectors get scaled (possibly by different scalars).

1.5 Summary of Technical Ideas

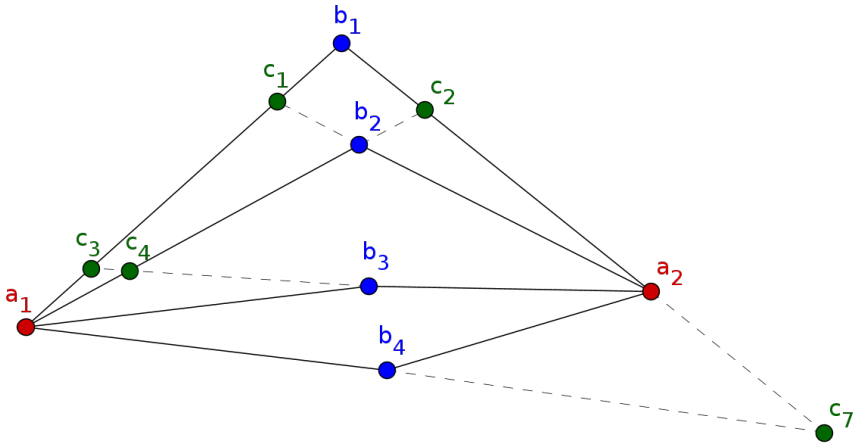
This Subsection includes the very broad technical ideas we used. First we explain a technique to reconstruct points from their projections. Then we give an overview of the Project-Reconstruct-Lift algorithm and how we plan to execute it. After that we illustrate the algorithm in quite generality. In this illustration we keep a lot of technicalities aside and try to motivate and picturize the algorithm through geometric intiuition.

1.5.1 A Simple Reconstruction Technique

We describe a method to recover points from their projections. A more rigorous treatment is in Appendix C. It also contains details and proofs of the Algorithm that is used in this paper. Suppose we have two disjoint sets of points $A = \{a_1, a_2\}, B = \{b_1, \ldots, b_d\}$ in the projective space $\mathbb{P}^{n+1}$ such that:

- We know the set A.
- We know the projections of points in B w.r.t. a_1 and a_2 i.e we know lines joining $L_{i,j} = \overrightarrow{a_i, b_j}$ for $i \in [2]$ and $j \in [d]$.

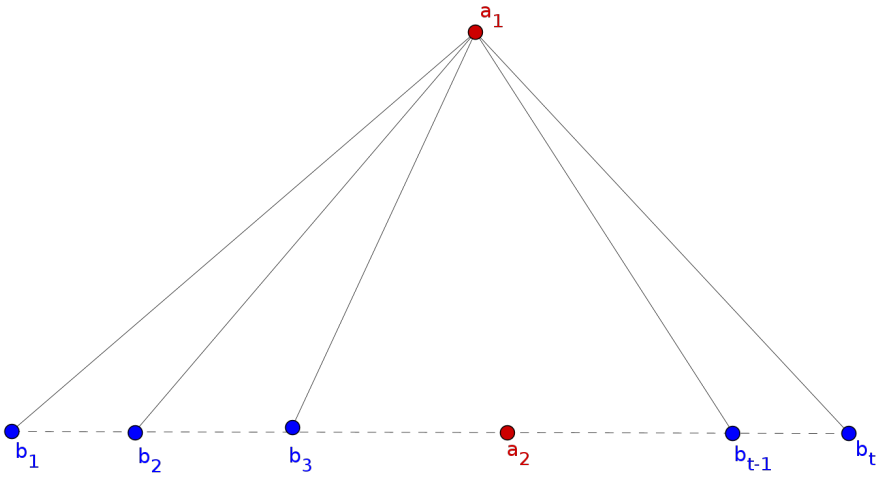
We want to use lines $L_{i,j} = \overrightarrow{a_i b_j}$ to find the set $\{b_1, \ldots, b_t\}$ in $O(poly(d))$ time. Note that there are $\leq d$ lines through a_1 and $\leq d$ lines through a_2. The b_j's lie at the intersection of these lines and so we have $\leq d^2$ intersections. These intersections form a set of candidate points for B but it is very hard to cutdown this set to B in $poly(d)$ time. There is a trivial $O(\binom{d^2}{d})$ algorithm - Go through all d points in these intersection points, make the lines and check if you get the same set of lines. This will give all sets of size d which could generate this configuration. Here is how the entire point configuration looks like. The green points c_j's are intersections of our lines which do not belong to B.



However if one assumes some restrictions then a subset of B might be found in $poly(d)$ time. Assume that for some $t \in [d]$:

- $\{a_1, a_2, b_1\}$ are affinely independent.
- $fl\{a_2, b_1\} \cap B = \{b_1, \ldots, b_t\}$.
- $fl\{a_1, a_2, b_1\} \cap B = \{b_1, \ldots, b_t\}$.

That is we have a sub configuration that looks like:



Here is an algorithm to recover all $\{b_1, \ldots, b_t\} \subset B$ such that the above conditions are satisfied.

- We iterate through all lines passing through a_2.
- For each such line L, find the set of lines S_L through a_1 which intersects L. Clearly all lines in S_L and L are co-planar.
- If this plane does not contain any other line through a_2, output the intersections of lines in S_L with L.

It is more or less straightforward that this algorithm works. The line L we choose has to have some b_j on it. Now all lines $\tilde{L} \in S_L$ that intersect L have to intersect it in some b_i otherwise $\tilde{L}$ has some other b_s on it but then the plane of S_L, L will have another line $\overrightarrow{a_2 b_s}$ passing through a_2 on it which is a contradiction. The algorithm actually finds all such configurations $\{b_1, \ldots, b_t\} \subset B$.

1.5.2 General Overview of the Algorithm

The broad structure of our algorithm is similar to that of Shpilka in [24] however our techniques are different. We first restrict the blackbox inputs to a low ($O(1)$) dimensional random subspace of $\mathbb{F}^n$ and interpolate this restricted polynomial. Next we try to recover the $\Sigma\Pi\Sigma(2)$ structure of this restricted polynomial and finally lift it back to $\mathbb{F}^n$. The random subspace and unique $\Sigma\Pi\Sigma(2)$ structure will ensure that the lifting is unique. Similar to [24] we try to answer the following questions. However our answers (algorithms) are different from theirs:

1. For a $\Sigma\Pi\Sigma(2)$ polynomial f over $r = O(1)$ variables, can one compute a small set of linear forms which contains all factors from both gates?

2. Let V_0 be a co-dimension k subspace($k = O(1)$) and $V_1, \ldots, V_t$ be co-dimension 1 subspaces of a linear space V. Given circuits C_i ($i \in \{0, \ldots, t\}$) computing $f|_{V_i}$(restriction of f to V_i) can we reconstruct from them a single circuit C for $f|_V$?
3. Given co-dimension 1 subspaces $V \subset U$ and circuits $f|_V$ when is the $\Sigma\Pi\Sigma(2)$ circuit representations of lifts of $f|_V$ to $f|_U$ unique?

Our first question is easily solved using Brill's equations (See Chapter 4 [8]). These provide a set of polynomials whose simultaneous solutions completely characterize coefficients of complex $\Pi\Sigma$ polynomials. A linear form $l = x_1 - a_2x_2 - \ldots - a_rx_r$ divides one of the gates of $f(x_1, \ldots, x_r) \Rightarrow f(a_2x_2 + \ldots + a_rx_r, x_2, \ldots, x_r)$ is a $\Pi\Sigma$ polynomial modulo l. When this is applied into Brill's equation (see Corollary A.2) we recover possible l's which obviously include linear factors of gates. We can show that (see Claim E.2) the extra linear forms we get are not too many ($poly(d)$) and also have some special structure. We call this set $\mathcal{C}$ of linear forms as Candidate linear forms and non-deterministically guess from this set. It should be noted that we do all this when our polynomial is over $O(1)$ variables.

We deal with the second question while trying to reconstruct the $\Sigma\Pi\Sigma(2)$ representation of the interpolated polynomial $f|_V$, where V is the random low dimensional subspace. We divide the algorithm into Easy Case, Medium Case and a Hard Case.

- For the Easy Case our algorithm tries to reconstruct one of the multiplication gates of $f|_V$ by first looking at it's restriction to a special co-dimension 1 subspace V_1. If $f = A + B$ with A, B being $\Pi\Sigma$ polynomials, the projection of one of the gates (say A) with respect to V_1 will be 0 and the other (say B) will remain unchanged giving us B and therefore both gates by factoring $f|_V - B$.
- In the Medium Case we have alteast two extra dimensions in one of the gates. This can be used to show that the only linear factors of $f_{|V}$ are those coming from G. Now we can recover G by factoring f and then use Easy Case for the remaining polynomial. An important consequence of this case is that in the Hard Case we may now assume that both gates are high dimensional which is very crucial.
- In the Hard Case we will first need V_0, a co-dimension k (where $k = O(1)$) subspace and then iteratively select co-dimension 1 subspaces $V_1, \ldots, V_t$. For some gate (say B), all pairs (V_0, V_i) ($i \in [t]$) will reconstruct some linear factors of B. This process will either completely reconstruct B or we will fall into the Easy Case. Once B is known we can factor $f|_V - B$ to get A.

The restrictions that we compute always factor into product of linear forms and can be easily computed since we know $f|_V$ explicitly. They can then be factorized into product of linear forms using the factorization algorithms from [14]. It is the choice of the subspaces $V_0, V_1, \ldots, V_t$ where our algorithm differs from that in [24] significantly. Our algorithm selects V_0 and iteratively selects the V_i's ($i \in [t]$) such that (V_0, V_i) have certain "*nice*" properties which help us recover the gates in $f|_V$. The existence of subspaces with "*nice*" properties is guaranteed by Quantitative Sylvester Gallai Theorems given in [3]. To use the theorems we had to develop more machinery that has been explained later.

The third question comes up when we want to lift our solution from the random subspace V to the original space. This is done in steps. We first consider random spaces U such that V has co-dimension 1 inside them. Now we reconstruct the circuits for $f|_V$ and $f|_U$. The $\Sigma\Pi\Sigma(2)$ circuits for $f|_V$ and $f|_U$ are unique since the simple ranks are high enough (because U, V are random subspaces of high enough dimension) implying that the circuit for $f|_V$ lifts to a unique circuit for $f|_U$. When this is done for multiple U's we can find the gates exactly.

1.5.2.1 Project-Reconstruct-Lift Algorithm

Here is a broad outline of the three aspects. This technique is quite common. Details of Project and Lift are in Section 4 and that of Reconstruct is in Section 3.

1.5.2.2 Project

- Input:$f \in \mathbb{F}[x_1, \ldots, x_n]$ as blackbox
- Choose random basis $\{y_1, \ldots, y_n\}$ of $\mathbb{F}^n$, $V = sp(\{y_1, \ldots, y_s\}), V_i = sp(\{v_1, \ldots, v_s, v_i\})$ for $i \in \{s+1, \ldots, n\}$.
- Define $f_0(y_1, \ldots, y_s) = f_{|V}, f_i(y_1, \ldots, y_s, y_i) = f_{|V_i}$.
- Consider sets $H \subset V, H_i \subset V_i$ with $|H| \geq d^s, |H_i| \geq d^{s+1}$ and interpolate to find f_0, f_i.

1.5.2.3 Reconstruct

- Reconstruct to get $f_0 = M_0 + M_1$ and $f_i = M_0^i + M_1^i$ with $M_0, M_1 \in \Pi\Sigma[y_1, \ldots, y_s], M_0^i, M_1^i \in \Pi\Sigma[y_1, \ldots, y_s, y_i]$.

1.5.2.4 Lift

- Use M_0, M_1, M_0^i, M_1^i to compute gates N_0, N_1 such that $f = N_0 + N_1$.
- If the reconstruction was successful return it, else return failed.

2 An Illustrative Example

Let $\bar{x}$ denote the variables $(x_1, \ldots, x_r)$ where r is a constant (we will fix this constant later). Consider the following polynomial $f(\bar{x}) \in \mathbb{F}[x_1, \ldots, x_r]$

$$f(\bar{x}) = T_0(\bar{x}) + T_1(\bar{x})\,.$$

Such that:

1. $T_0(\bar{x}) = A_1 \ldots A_d, T_1(\bar{x}) = B_1 \ldots B_d$ with A_i, B_j linear forms
2. $gcd(A_i, B_j) = 1$ for all $1 \leq i, j \leq d$.
3. $dim(\{A_i, B_j : i, j \in [d]\}) = r$ i.e. there are no redundant variables.

Define the sets $A = \{A_1, \ldots, A_d\}$, $B = \{B_1, \ldots, B_d\}$. We are going to view the points in A and B as points in the space $\mathbb{F}^r$. We also identify (keep only one copy) linear forms which are scalar multiples of each other.

▶ **Theorem 2.1.** *Consider $f(\bar{x})$ from above and assume $f(\bar{x}) = \sum_{\lambda \in \Lambda} \mathbf{c}_\lambda \mathbf{x}^\lambda$ where $\lambda = (\lambda_1, \ldots, \lambda_r)$ and $\mathbf{x}^\lambda = x_1^{\lambda_1} \ldots x_r^{\lambda_r}$. Suppose we know all the coefficients $\mathbf{c}_\lambda$ then in time poly(d) we can reconstruct $T_0(\bar{x}), T_1(\bar{x})$ with high probability.*

We will describe an algorithm which proves the above theorem. At many points during the algorithm we will need results that are mentioned later in the paper. For better understanding we encourage the reader to first go through this algorithm assuming all the claims mentioned.

2.1 Candidate Linear Forms

Our job in this algorithm is to reconstruct $T_0(\bar{x}), T_1(\bar{x})$ i.e. A_i's and B_j's. Let us first observe a property these linear forms satisfy. One can see that for $l \in \{A_i, B_j : i, j \in [d]\}$ the following holds:

$f_{|l=0}$ is a non-zero product of linear forms .

Can we use this to reconstruct A_i, B_j? The two questions that pop up are:

1. Are there linear forms other than A_i, B_j that satisfy the above condition?
2. If yes, can we find out some structure of the bad l's (which are not A_i, B_j)?
3. Can we bound the total number of such l's by a polynomial in d?
4. Can we construct this set efficiently?

The answer to all the above questions is a YES!

▶ **Example 2.2.** Consider $f(x_1, \ldots, x_r) = (x_1 + x_2)(x_1 + x_3) \ldots (x_1 + x_r) + x_2 \ldots x_r$. We can see that $f_{|x_1=0} = x_2 \ldots x_r$ but x_1 is not a factor of any of the gates.

The next claim contains the information structure of the bad l's and their number. Proof will be given later in the paper in Appendix E.

▶ **Claim 2.3.** *Consider the set $\mathcal{C} = \{l : f_{|l=0}$ is a non zero product of linear forms $\}$ and let $\{l_1, \ldots, l_k\} \subset T_i$ be a set of LI linear forms where $k = c_{\mathbb{F}}(3) + 2$ (rank bound for $\Sigma\Pi\Sigma(3)$ circuits) then*

1. $\{A_i, B_j : i, j \in [d]\} \subseteq \mathcal{C}$
2. $|\mathcal{C}| \leq O(d^4)$
3. *If $l \in \mathcal{C} \setminus \{A_i, B_j, i, j \in [d]\}$, then there exists $i \in [k]$ and $j \in [d]$ such that $\{l, A_i, B_j\}$ are linearly dependent i.e. for every LI set $\{A_1, \ldots, A_k\}$, a bad l will match one of these A_i ($i \in [k]$) to some B_j.*

Moreover the above set $\mathcal{C}$ can be constructed in time $poly(d)$. This is done by solving a set of multivariate polynomial equations of $poly(d)$ degree in $O(1)$ variables. Please see Appendix E for details.

2.2 Reconstruction Algorithm

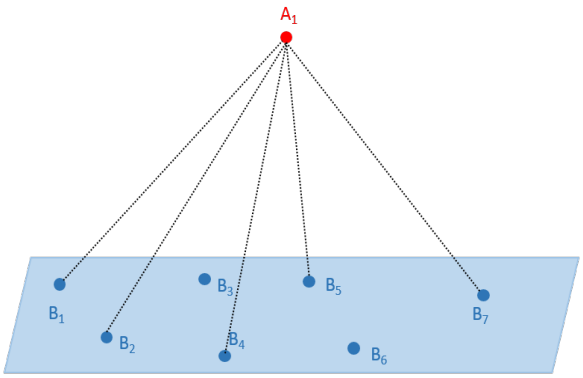
Before going to the core of the algorithm let's explain an easy case. Recall $A = \{A_1, \ldots, A_d\}$ and $B = \{B_1, \ldots, B_d\}$. Also color the points in A red and the points in B blue.

2.2.1 Easy Case

For this case we assume

$$\boxed{sp(A) \subsetneq sp(B)\,.}$$

So let's say $A_1 \notin sp(B)$. The main advantage of such an A_1 is that on setting A_1 to 0 no linearly independent $\{B_i, B_j\}$ become dependent. Geometrically we have the following picture:



We guess a basis $\{l_1, \ldots, l_r\}$ of linear forms from the set $\mathcal{C}$. While doing this we assume:

- $l_1 = A_1$
- $l_2, \ldots, l_t$ is a basis for B
- $l_{t+1}, \ldots, \ldots, l_r$ are the rest of the basis vectors

If our guess was actually a basis we define an invertible linear transformation T sending l_i to x_i. We apply T to $f(\bar{x})$ by applying it to each variable in the most natural way. If our guess was correct we get

$$f'(\bar{x}) = f(T(\bar{x})) = x_1 A'_2 \ldots A'_d + B'_1 \ldots B'_d \,.$$

Note that if our assumption for the basis is correct then none of the B'_i's contain x_1. So we can compute $f'_{|_{x_1=0}} = B'_1 \ldots B'_d$. Then we can apply T^{-1} and get back $T_1(\bar{x}) = B_1, \ldots, B_d$. We remind the reader that everything is recovered upto a scalar multiple but that is not a problem since that can be merged into one scalar for the gate $B(\bar{x})$ which can be easily recovered. We then factorize $f - T_1(\bar{x})$ and check if it factors into a product of linear forms and recover $T_0(\bar{x})$. Note that during the process we will guess the basis correctly atleast once. Also the last step checks if we actually get a $\Sigma\Pi\Sigma(2)$ circuit and therefore the reconstruction will be complete. The case where $sp(B) \subsetneq sp(A)$ is symmetrical and is handled in the same way. Next we deal with the hard case.

2.2.2 Hard Case

The other case i.e. $sp(A) = sp(B)$ is much harder but high dimensionality enables us to apply the Quantitative version of Sylvester Gallai Theorems from [3]. Let's first just give some consequences of the Quantitative Sylvester Gallai theorem (from [3]) which will be useful for us. A slightly more general version with proof can be found in Appendix B.

▶ **Corollary 2.4.** *Let $S = \{s_1, \ldots, s_n\} \subseteq \mathbb{C}^d$ be a set of points. Assume $dim(S) > \Omega(C^k)$ for some constant C, then there exists a set of linearly independent points $\{s_1, \ldots, s_k\}$ and a set $T \subset S$ with $|T| \geq 0.99n$, such that $fl(\{s_1, \ldots, s_k, t\})$ is an elementary $k-flat$ for every $t \in T$. That is:*

- $t \notin fl(\{s_1, \ldots, s_k\})$
- $fl(\{s_1, \ldots, s_k, t\}) \cap S = \{s_1, \ldots, s_k, t\}$.

▶ **Lemma 2.5** (Bichromatic semi-ordinary line)**.** *Let X and Y be disjoint finite sets in $\mathbb{C}^d$ satisfying the following conditions.*

1. *$dim(Y) > \Omega(C^4)$ where C is the constant in the above corollary.*
2. *$|Y| \leq 99|X|$*

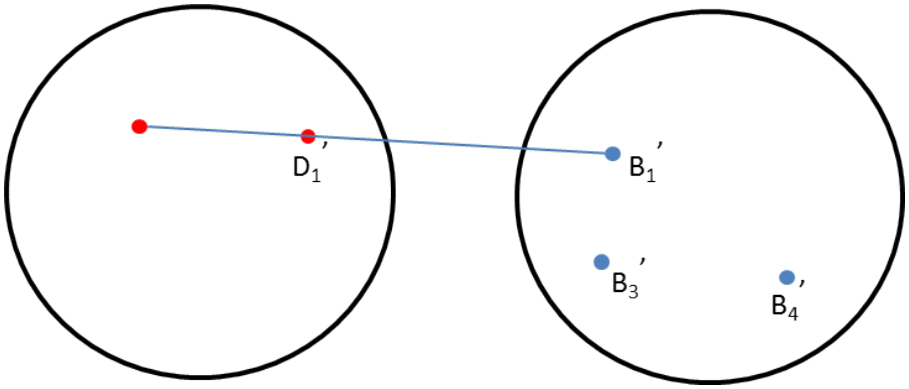
Then there exists a line l such that $|l \cap Y| = 1$ and $|l \cap X| \geq 1$.

At this point we would like to mention that the constants $99, 0.99$ and the one hidden in $\Omega(C^k)$ have more general values given by a parameter δ. For the time being we've fixed them for better exposition. Please see Appendix B for more details.

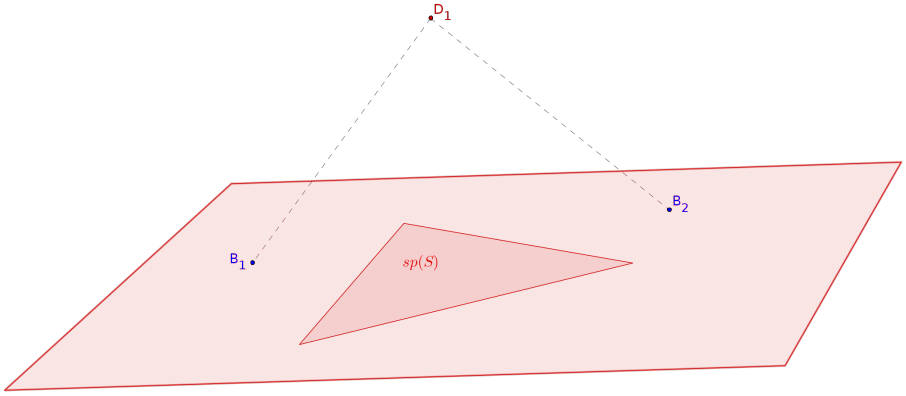
Using high dimensionality of A, B and the above mentioned corollaries we are able to prove the following theorem which forms the backbone of our algorithm.

▶ **Theorem 2.6.** *For some product gate (say A), there exists $k = O(1)$ points $S = \{A_1, \ldots, A_k\}$ and a large set $D \subset A$ such that on projecting D, B to the subspace W defined by $\{A_1 = 0, \ldots, A_k = 0\}$ (and throwing away zeros):*

- *There exists a lines $L = \overrightarrow{B_1' D_1'}$ where B_1' and D_1' are projections of B_1, D_1 onto W. Also if B' is the projection of B onto W then $L \cap B' = \{B_1'\}$, so the line is a bichromatic semi-ordinary which were discussed in the lemma above.*



Let's pick one of these lines and see what would have happened in $\mathbb{F}^r$ which led us to this line in W.



In the picture above the inner triangle denotes $sp(S)$ and the outer parallelogram denotes $sp(S \cup \{B_1\})$. The line in the previous picture i.e. projecting the points onto W has only one blue point implying:

- $sp(S \cup \{B_1\}) \cap B = sp(S \cup \{B_1\})$
- $sp(S \cup \{B_1\} \cup \{D_1\}) \cap B = sp(S \cup \{B_1\})$

Note that this looks very similar to what we had in Subsection 1.5.1. We used this kind of a configuration to recover points using their projections. A similar method is implemented here. Given that such a configuration exists we can come up with the following algorithm.

1. From the set $\mathcal{C}$ guess the set $S = \{A_1, \ldots, A_k\}$ mentioned in the theorem above.
2. Using condition (3) in Claim 2.3 obtain a set X such that $D \subset X \subset A$. This can be done as explained in Algorithm 4. The reader should just assume this at the moment. We need to make sure that D_1 comes from A because the algorithm is iterative and we don't want a spurious linear form in $\mathcal{C}$ give any reconstruction. We always want to set some A_i's to 0 so that we only recover B_j's.
3. Iterate over this set X and guess D_1.
4. By projecting f to the subspaces $\{A_1 = 0, \ldots, A_k = 0\}$ and $\{D_1 = 0\}$ we get B_1' and $(B_1)_{|_{D_1=0}}$. Because of the diagram above these two projections can be matched and used to reconstruct B_1.
5. If no $D_1 \in X$ worked then go to Easy Case since dimension should have fallen.

Basically the algorithm just exploits the existence of the line mentioned in the previous theorem and reconstructs the corresponding B_1 (whose projection lies on the line). This reconstruction was possible because this line had only one blue point. After finding B_1 we declare it known so that in the next iteration we can remove it's projection when required. We will continue to get such bichromatic semi-ordinary lines till the unkown linear forms in the B set have high dimension. If at any stage this reconstruction is not possible then this dimension would have fallen and we can use the Easy Case.

2.2.2.1 Return Type

In all our algorithms we wish to return the reconstructed form of f. Since f and the two gates T_0, T_1 are to be returned we define an object for it. We call this object Decomposition. We assume having a data type polynomial for general polynomials and pi_sigma for polynomials which are product of linear forms. We use C++ syntax to define our structure.

```
struct decomposition {
  bool iscorrect; // iscorrect will be true if f = M_0 + M_1
  polynomial f;
  pi_sigma M_0;
  pi_sigma M_1;

  // Constructor when a reconstruction is found
  decomposition(polynomial g, pi_sigma A, pi_sigma B){
    iscorrect =true;
    f=g;
    M_0=A;
    M_1=B;
  }

  // Constructor when no reconstruction is found
  decomposition(){
    iscorrect=false;
  }
};
```

3 Reconstruction for low rank

Let's recall Definition 1.2 following Theorem 1.1 in Section 1.

▶ **Definition 3.1.** We fix s to be any constant $> \max(C_{2k-1} + k, c_{\mathbb{F}}(4))$ where:

1. $c_{\mathbb{F}}(l) = 3l^2$ is the rank lower bound (see Theorem 1.7) that guarantees non-zeroness of any simple, minimal, $\Sigma\Pi\Sigma(l)$ circuit with rank $> c_{\mathbb{F}}(l)$.

2. $k = c_{\mathbb{F}}(3) + 2$.
3. δ is some fixed number in $(0, \frac{7-\sqrt{37}}{6})$.
4. $C_k = \frac{C^k}{\delta}$ the constant that appears in Theorem B.4.

Let r be any constant $\geq s$ (In our application we need s and $s+1$). Our main theorem for this section therefore is:

▶ **Theorem 3.2.** *Let r be as defined above. Consider $f(\bar{x}) \in \mathbb{F}[\bar{x}]$, a multivariate homogeneous polynomial of degree d over the variables $\bar{x} = (x_1, \ldots, x_r)$ which can be computed by a $\Sigma\Pi\Sigma(2)$ circuit C over $\mathbb{F}$. Assume that rank of the simplification of C i.e. $Sim(C) = r$. We give a $poly(d)$ time randomized algorithm which computes C given blackbox access to $f(\bar{x})$.*

We assume f has the following $\Sigma\Pi\Sigma(2)$ representation:

$$f = \tilde{G}(\tilde{\alpha}_0\tilde{T}_0 + \tilde{\alpha}_1\tilde{T}_1)$$

where $\tilde{G}, \tilde{T}_i \in \Pi\Sigma_{\mathbb{F}}[\bar{x}]$ are *normal* (i.e. leading non-zero coefficient is 1 in every linear factor) and $\tilde{\alpha}_0, \tilde{\alpha}_1 \in \mathbb{F}$ with $gcd(\tilde{T}_0, \tilde{T}_1) = 1$. The $rank(Sim(C)) = r$ condition then becomes

$$sp(\mathcal{L}(\tilde{T}_0) \cup \mathcal{L}(\tilde{T}_1)) = Lin_{\mathbb{F}}[\bar{x}]\,.$$

Consider the set $T = \mathcal{L}(\tilde{G}) \cup \mathcal{L}(\tilde{T}_0) \cup \mathcal{L}(\tilde{T}_1)$. By abuse of notation we will treat these linear forms also as points in $\mathbb{F}^r$. Since linear factors of $\tilde{G}, \tilde{T}_i$ are normal, two linear factors of $\tilde{G}, \tilde{T}_i$ are LD iff they are same.

Random Transformation and Assumptions

Let Ω, Λ be two $r \times r$ matrices such that their entries $\Omega_{i,j}$ and $\Lambda_{i,j}$ are picked independently from the uniform distribution on $[N]$. Here $N = 2^d$. We begin our algorithm by making a few assumptions. All of these assumptions are true with very high probability and we assume them in our algorithm. These assumptions make our work easy by removing redundancy in the co-ordinates. *The idea is to move vectors randomly thereby introducing non-zero coefficients in them.* Consider the standard basis of $\mathbb{F}^r$ given as $\mathcal{S} = \{e_1, \ldots, e_r\}$. Let $E_j = sp(\{e_1, \ldots, e_j\})$ and $E'_j = sp(\{e_{j+1}, \ldots, e_r\})$, clearly $\mathbb{F}^r = E_j \oplus E'_j$. Let $\pi_{W_{E_j}}$ be the orthogonal projection onto E_j w.r.t. this decomposition. Note that T is a finite set of vectors in $\mathbb{F}^r$.

- **Assumption 0:** Ω is invertible. This is just the complement of event $\mathcal{E}_0$ in Section D and so occurs with high probability.
- **Assumption 1:** For all $t \in T$, $\pi_{W_{E_1}}(\Omega(t)) \neq 0$ i.e. $[\Omega(t)]^1_{\mathcal{S}} \neq 0$ (coefficient of e_1 is non-zero). This is the complement of event $\mathcal{E}_1$ in Section D. and so occurs with high probability.
- **Assumption 2:** For all LI sets $\{t_1, \ldots, t_r\} \subset T$, $\{\Omega(t_1), \ldots, \Omega(t_r)\}$ is LI. This essentially means that Ω is invertible. This is the complement of $\mathcal{E}_2$ in Section D and so occurs with high probability.
- **Assumption 3:** Fix a $k < r$. For all LI sets $\{t_1, \ldots, t_r\} \subset T, \{\Omega(t_1), \ldots, \Omega(t_k), \Lambda\Omega(t_{k+1}), \ldots, \Lambda\Omega(t_d)\}$ is LI i.e. is a basis. This is the complement of event $\mathcal{E}_3$ in Section D and so occurs with high probability. It'll be used later in this paper.
- **Assumption 4:** Fix a $k < r$. For all LI sets $\tilde{T} = \{t_1, \ldots, t_r\} \subset T$, define the set $\mathcal{B} = \{\Omega(t_1), \ldots, \Omega(t_k), \Lambda\Omega(t_{k+1}), \ldots, \Lambda\Omega(t_r)\}$. By Assumption 3 this is a basis. Consider any $t \in T$ such that $\Omega(t) \notin sp(\{\Omega(t_1), \ldots, \Omega(t_k)\})$. Then $[\Omega(t)]^{k+1}_{\mathcal{B}} \neq 0$. This event is the complement of $\mathcal{E}_5$ and so it occurs with high probability. We want non-zeroness of co-ordinates even after projecting to a codimension-k subspace. That is where this will be useful.

From now onwards we will assume that all the above assumptions are true. Since all of them occur with very high probability, their complements occur with very low probability and by union bound the union of their complements is a low probability event. So intersection of the above assumptions occurs with high probability and we assume all of them are true. *Note that the assumptions will continue to be true if we scale all linear forms (possibly different scaling for different vectors, but non-zero scalars) in T i.e. if the assumptions were true for T then they would have been true had we started with a scaling of T.*

The first step of our algorithm is to apply Ω to f. We have a natural identification between linear forms and points in $\mathbb{F}^r$. This identification converts Ω into a linear map on $Lin_{\mathbb{F}}[\bar{x}]$ which can be further converted to a ring homomorphism on polynomials by assuming that it preserves the products and sums of polynomials. So Ω gets applied to all linear forms in the $\Sigma\Pi\Sigma(2)$ representation of f. Since f is a degree d polynomial in r variables it has atmost $poly(d^r)$ coefficients. Applying Ω to each monomial and expanding it takes $poly(d^r)$ time and gives $poly(d^r)$ terms. So computing $\Omega(f)$ takes $poly(d^r)$ time and has $poly(d^r)$ monomials.

Now we try and reconstruct the circuit for $\Omega(f)$. If this reconstruction can be done correctly, we can apply Ω^{-1} and get back f. Note that Assumption 1 tells us that the coefficient of x_1 in $\Omega(l)$ is non-zero for all l in T. Let $X = \{x_1, \ldots, x_r\}$ and $\bar{x}$ is used for the tuple $(x_1, \ldots, x_r)$. From this discussion we know that:

$$\Omega(f) = \Omega(\tilde{G})(\tilde{\alpha}_0\Omega(\tilde{T}_0) + \tilde{\alpha}_1\Omega(\tilde{T}_1)) = G(\alpha_0 T_0 + \alpha_1 T_1)$$

where α_i are chosen such that linear factors of G, T_i have their first coefficient(that of x_1) equal to 1. So they are *standard* $\Pi\Sigma$ polynomials. Note that we've used Assumption 1 here. Since we've moved constants to make linear forms standard we can assume $G = \lambda\Omega(\tilde{G}), T_i = \lambda_i\Omega(\tilde{T}_i)$ with $\lambda, \lambda_i \in \mathbb{F}$. Consider some scaling T_{sc} of T such that $\mathcal{X} = \mathcal{L}(G) \cup \mathcal{L}(T_0) \cup \mathcal{L}(T_1)$ is $= \Omega(T_{sc})$. All above assumptions are true for T_{sc} and so we may use the conclusions about $\Omega(T_{sc})$ i.e. $\mathcal{X}$. Also since Ω is invertible $gcd(T_0, T_1) = 1$.

Let

$$T_i = \prod_{j\in[M]} l_{ij}, i = 0, 1 \text{ and } G = \prod_{k\in[d-M]} G_k$$

with l_{ij}, G_k linear forms (so $d = deg(f)$).

For simplicity from now onwards we call $\Omega(f)$ by f and try to reconstruct it's circuit. Once this is done we may apply Ω^{-1} to all the linear forms in the gates and get the circuit for f. This step clearly takes $poly(d^r)$ time in the same way as applying Ω took. Since r is a constant, the steps described above take $poly(d)$ time overall.

Known and Unknown Parts

We also define some other $\Pi\Sigma$ polynomials $K_i, U_i, i = 0, 1$ which satisfy

$$K_i \mid \alpha_i G T_i, U_i = \frac{\alpha_i G T_i}{K_i}.$$

with the extra condition

$$gcd(K_i, U_i) = 1.$$

K_i are the known factors of $\alpha_i GT_i$ and U_i the unknown factors. The *gcd* condition just means that that known and unknown parts of $\alpha_i GT_i$ don't have common factors. In other words linear forms in $\alpha_i GT_i$ are known with full multiplicity. We initialize $K_i = 1$ and during the course of the algorithm update them as and when we recover more linear forms. At the end $K_i = \alpha_i GT_i$ and so we know both gates.

3.1 Outline of the algorithm

1. **Set $\mathcal{C}$ of Candidate Linear Forms:** We compute a $poly(d)$ size set $\mathcal{C}$ of linear forms which contains $\mathcal{L}(T_i), i = 0, 1$. We will non-deterministically guess from this set $\mathcal{C}$ making only a constant number of guesses everytime(thus polynomial work overall). It is important to note that the uniqueness of our circuit guarantees that our answer if computed can always be tested to be right. For more details on this please see Appendix E. We also give an efficient algorithm to construct this set. See Algorithm 8.
2. **Easy Case:** $\boxed{\mathcal{L}(T_{1-i}) \subsetneq sp(U_i), \text{ for some } i \in \{0, 1\}}$
   So T_{1-i} has a linear factor $l_{(1-i)1}$ such that

   $$sp(\{l_{(1-i)1}\}) \cap sp(U_i) = \{0\} \tag{1}$$

   Let $W = sp(\{l_{(1-i)1}\})$ and extend to a basis of V and in the process obtain another subspace $W' \subset V$ such that $W \oplus W' = V$. We can see from Equation 1 that LI linear forms in U_i remain LI when we project to W'. We use this to compute U_i and then since $K_i U_i = \alpha_i GT_i$ we know one of the gates. To find the other gate simply factorize $f - \alpha_i GT_i$. If it factors into a product of linear forms we have the reconstruction.
3. **Medium Case:** $\boxed{dim(^{sp(T_{1-i}) + sp(T_i)}/_{sp(T_i)}) \geq 2 \text{ for some } i \in \{0, 1\}}$
   This case is just to facilitate the Hard Case. We know that T_{1-i} has two linear factors $l_{(1-i)1}, l_{(1-i)2}$ such that $sp(\{l_{(1-i)1}, l_{(1-i)2}\}) \cap sp(T_i) = \{0\}$. We show that the only linear factors of f are those which appear in G. So we can first factorize f using Kaltofen's factoring ([14]) and obtain G. Update $K_j = G, j = 0, 1$. So $U_j = \alpha_j T_j$ for $j = 0, 1$. Clearly we also have $\mathcal{L}(T_{1-i}) \subsetneq sp(T_i) = sp(U_i)$ and we can go to Easy Case above with $K_i = G$.
4. **Hard Case:** $\boxed{\mathcal{L}(T_{1-i}) \subseteq sp(U_i), \text{ for } i = 0 \text{ and } 1}$
   We know that we are not in Medium Case and so $dim(sp(T_0) + sp(T_1)) - sp(T_i) \leq 1$ for $i = 0, 1$. Also $dim(sp(T_0) + sp(T_1)) = r$ by assumption on the simple rank of our polynomial. So this guarantees that $dim(sp(T_{1-i})) \geq r - 1 \Rightarrow$ (by the condition of this hard case) $dim(sp(U_i)) \geq r - 1$ for $i = 0, 1$. This enables us to use the Quantitative Sylvester Gallai theorems on both sets $\mathcal{L}(T_i), \mathcal{L}(U_i)$.
   - Our first step is to identify a certain "*bad*" $\Pi\Sigma$ factor I of G and get rid of it to get $G = \frac{G}{I}$ and thus $f = \frac{f}{I}$. The factors of I don't satisfy certain properties we need later and so we remove them. Thankfully we have an efficient algorithm to recover I. Our algorithm uses something we call a Detector Pair (See 3.4) whose existence is shown using the Quantitative Sylvester Galai Theorems mentioned above.
   - So now our job is to reconstruct f with known (and unknown resp.) parts as $K_0^\star, K_1^\star$ ($U_0^\star, U_1^\star$ resp.).
   - If $sp(U_{1-i})$ becomes low dimensional we may fall in Easy Case and recover the circuit for f directly. Otherwise the same detector pairs then provide certain "*nice*" subspaces corresponding to linear forms in T_i. Projection of U_{1-i} onto these subspaces can be easily glued together to recover some linear factors(with multiplicities) of U_{1-i}, which will then be multiplied to $K_{1-i}^\star$.

- The process continues as long as $sp(U_{1-i})$ remains high dimensional. As soon as this condition fails we end up in Easy Case and the gates are recovered.

We give algorithms for Easy and Medium cases. Hard Case will require more prepration and will be done after these subsections. From now onwards we assume that we have constructed a $poly(d)$ sized set of linear forms $\mathcal{C}$ which contains $\mathcal{L}(T_i)$ for $i = 0, 1$. We have other structural results about linear forms in this set. See Appendix E for more details and algorithms. Algorithm 8 constructs this set in $poly(d)$ time.

3.2 Easy Case

$$\boxed{\mathcal{L}(T_{1-i}) \subsetneq sp(U_i), \text{ for some } i \in \{0,1\}}$$

▶ **Claim 3.3.** *Suppose for some $i \in \{0,1\}$, $\mathcal{L}(T_{1-i}) \subsetneq sp(U_i)$ then we can reconstruct f.*

FunctionName : EasyCase
input : $f \in \Sigma\Pi\Sigma_{\mathbb{F}}(2)[\bar{x}], K_0 \in \Pi\Sigma_{\mathbb{F}}[\bar{x}], K_1 \in \Pi\Sigma_{\mathbb{F}}[\bar{x}], \mathcal{C} \subset Lin_{\mathbb{F}}[\bar{x}]$)
output : An object of type *decomposition*

for $i \leftarrow 0$ **to** 1 **do**
  for *each LI set* $\{l_1, l_2, \ldots, l_r\} \subset \mathcal{C}$ **do**
    Define $K_i' \leftarrow K_i$;
    Find t such that $l_1^t \mid\mid f$;
    // i.e. $l_1^t \mid f$ && $l_1^{t+1} \nmid f$
    $W \leftarrow sp(\{l_1\}), W' \leftarrow sp(\{l_2, \ldots, l_r\})$;
    if $l_1^t \mid\mid K_i'$ **then**
      $\tilde{f} = \frac{f}{l_1^t}$; $\tilde{K_i} = \frac{K_i'}{l_1^t}$;
      if $U_i = \frac{\pi_{W'}(\tilde{f})}{\pi_{W'}(\tilde{K_i})} \in \Pi\Sigma_{\mathbb{F}}[\bar{x}]$ && $f - K_iU_i \in \Pi\Sigma_{\mathbb{F}}[\bar{x}]$ **then** $K_i = K_iU_i$, $K_{1-i} = f - K_iU_i$;
      return *decomposition*(f, K_0, K_1);
    end
  end
end
return *decomposition*();

Algorithm 1: Easy Case Reconstruction.

Explanation and Correctness Analysis

- The first for loop just guesses the gate with extra dimensions i.e. it's not contained in span of the unknown part of the other gate.
- If for some basis $\{l_1, \ldots, l_r\} \subset \mathcal{C}$ the algorithm actually computes a $\Sigma\Pi\Sigma(2)$ representation in the end then it ought to be correct since the last 'if' also checks if it is correct.
- If our guess for i is correct, we show that there exists a basis $\{l_1, \ldots, l_r\} \subset \mathcal{C}$ for which all conditions will be satisfied and we actually arrive at a $\Sigma\Pi\Sigma(2)$ representation in the end. Since $\mathcal{L}(T_{1-i}) \subsetneq sp(U_i)$ and $\mathcal{L}(T_{1-i}), \mathcal{L}(U_i) \subset \mathcal{C}$ there exists $l_1 \in \mathcal{L}(T_{1-i}) \setminus sp(U_i) \subset \mathcal{C}$. Choose a basis $\{l_2, \ldots, l_s\}$ of $sp(U_i)$, then $\{l_1, \ldots, l_s\}$ is an LI set. Now extend this to a basis $\{l_1, \ldots, l_s, l_{s+1}, \ldots, l_r\} \subset \mathcal{C}$ of V. We go over all choices of basis in $\mathcal{C}$ and will arrive at the right one.

- We initialize a dummy polynomial K_i' to represent K_i since we do not want to update K_i till we actually have a solution. Let's assume $l_1^t \mid\mid f$ i.e. $l_1^t \mid f$ and $l_1^{t+1} \nmid f$. We know $l_1 \mid T_{1-i} \Rightarrow l_1 \nmid T_i \Rightarrow l_1 \nmid \alpha_i T_i + \alpha_{1-i}T_{1-i}$. Therefore $l_1^t \mid\mid G \Rightarrow l_1^t \mid\mid \alpha_i GT_i = K_iU_i$. Also $l_1 \notin sp(U_i) \Rightarrow l_1 \nmid U_i$ thus $l_1^t \mid\mid K_i \Rightarrow l_1^t \mid\mid K_i'$. We remove l_1^t from both f, K_i' to get $\tilde{f}, \tilde{K_i}$. Let $W = sp(\{l_1\})$ and $W' = sp(\{l_2, \ldots, l_r\})$, therefore $V = W \oplus W'$. Note that since $l_1 \in \mathcal{L}(T_{1-i})$

  $$\pi_{W'}(\tilde{f}) = \pi_{W'}(U_i)\pi_{W'}(\tilde{K_i})$$

  Since $\pi_{W'}(\tilde{K_i}) \neq 0$, we get $\pi_{W'}(U_i) = \frac{\pi_{W'}(\tilde{f})}{\pi_{W'}(\tilde{K_i})}$. If $U_i = u_1 \ldots u_s$ with $u_j \in W'$, we see that $\pi_{W'}(U_i) = \pi_{W'}(u_1) \ldots \pi_{W'}(u_s) = u_1 \ldots u_s = U_i$. So we get U_i and hence $\alpha_i GT_i = K_iU_i$. Once $\alpha_i GT_i$ is known we factorize $f - \alpha_i GT_i$ to get $\alpha_{1-i}GT_{1-i}$. For the correct choice of our basis this will factorize completely into a $\Pi\Sigma$ polynomial. Now we update $K_i = K_iU_i$ and $K_{1-i} = f - K_iU_i$ and an object $decomposition(f, K_0, K_1)$. Throughout the algorithm we use Kaltofen's factoring [14] wherever necessary.
- If we were not able to find the $\Sigma\Pi\Sigma(2)$ representation then we return an object $decomposition()$.

Time Complexity

We can see above all loops run only $poly(d)$ many times. The most expensive step is choosing r vectors from $\mathcal{C}$. But recall that r is a constant and so this also takes only polynomial time in d. Other steps like factoring polynomials (using Kaltofen's factoring algorithm from [14]), taking projection onto known subspaces, divding by polynomials require $poly(d)$ time (r is a constant) as has been explained multiple times before.

3.3 Medium Case

$$\boxed{dim(^{sp(T_{1-i}) + sp(T_i)}/_{sp(T_i)}) \geq 2 \text{ for some } i \in \{0, 1\}}$$

▶ **Claim 3.4.** *If* $dim(^{sp(T_{1-i}) + sp(T_i)}/_{sp(T_i)}) \geq 2$ *then* $\mathcal{L}(\alpha_i T_i + \alpha_{1-i}T_{1-i}) = \phi$.

Proof. $dim(^{sp(T_{1-i}) + sp(T_i)}/_{sp(T_i)}) \geq 2 \Rightarrow$, there exists $l_1', l_2' \in \mathcal{L}(T_{1-i}) \setminus sp(T_i)$ be such that $dim(\{l_1', l_2'\} \cup \mathcal{L}(T_i)) = dim(\mathcal{L}(T_i)) + 2$. Assume there exist $l \in \mathcal{L}(\alpha_i T_i + \alpha_{1-i}T_{1-i})$.

$l \mid \alpha_i T_i + \alpha_{1-i}T_{1-i} \Rightarrow l \nmid T_i$ and $l \nmid T_{1-i}$ (since they are coprime)

$$0 \neq \alpha_i \prod_{j \in [M]} l_{ij} = -\alpha_{1-i} \prod_{j \in [M]} l_{(1-i)j} \pmod{\{l\}}.$$

Thus there exist $l_1, l_2 \in \mathcal{L}(T_i)$ and scalars $\gamma_j, \delta_j, j \in [2]$ such that $l = \gamma_j l_j + \delta_j l_j'$. Since $l \nmid T_0, l \nmid T_1$ we get γ_j, δ_j are non zero.

$\delta_1, \delta_2 \neq 0 \Rightarrow$,

$$l_1', l_2' \in sp(\{l\} \cup \mathcal{L}(T_i)) \Rightarrow dim(\{l_1', l_2'\} \cup \mathcal{L}(T_i)) \leq dim(\mathcal{L}(T_i)) + 1$$

which is a contradiction. So $\mathcal{L}(\alpha_i T_i + \alpha_{1-i}T_{1-i}) = \phi$.

Therefore the only linear factors of f are present in G, which can now be correctly found by using Kaltofen's algorithm [14] and identifying the linear factors. Update $K_j = G$ for $j = 0, 1$, therefore $U_j = T_j$. Also this case implies that $\mathcal{L}(T_{1-i}) \subsetneq sp(T_i) = sp(U_i)$, and so we can use Easy Case. ◀

So we have the following claim:

▶ **Claim 3.5.** *If the condition in Medium Case is true, the following algorithm reconstructs f, if there is a reconstruction.*

FunctionName : MediumCase
input : $f \in \Sigma\Pi\Sigma_{\mathbb{F}}(2)[\bar{x}], \mathcal{C} \subset Lin_{\mathbb{F}}[\bar{x}]$)
output : An object of type *decomposition*

$L \leftarrow Lin(f)$;
// Use Kaltofen's factoring from [14] to compute $Lin(f) \stackrel{def}{=}$ product of all linear factors of f
if $EasyCase(f, L, L, \mathcal{C}) \rightarrow iscorrect$ **then**
  return $EasyCase\ (f, L, L, \mathcal{C})$;
end
return $decomposition()$;

Algorithm 2: Medium Case Reconstruction

The above algorithm does exactly what has been explained in the preceeding paragraph. It works in $poly(d)$ time if EasyCase$(f, K_0, K_1, \mathcal{C})$ works in $poly(d)$ time. Kaltofen's factoring and all other steps are $poly(d)$ time.

Now we need to handle the Hard Case. This is quite technical and so we do some more preparation. We devise a technique to get rid of some factors of f to get a new polynomial f without destroying the $\Sigma\Pi\Sigma(2)$ structure. If Easy Case holds for f we stop there itself. Otherwise we will use combination of different subspaces of V, project f onto them and glue projections to get gates for f.

3.4 Detector Pair, Reducing Factors, Hard Case Preparation

Let's recall:

$$g = \frac{f}{G} = \alpha_0 T_0 + \alpha_1 T_1$$

We outline an approach to identify some factors of f. These factors will divide G but won't divide g. This is going to be useful in the Hard Case. The linear factors left after removing these identified factors will have very strong structural properties and so will be instrumental in reconstruction. The main tool in this identification is a pair (S, D) (defined below) inside one of the $\mathcal{L}(T_i)$'s. This pair will be called a *"Detector Pair"*. It will also decide the subspaces on which we take projections of f and glue back to get the gates.

Detector Pairs (S, D)

Fix $k = c_{\mathbb{F}}(3) + 2$ (See Theorem 1.7 for definition of $c_{\mathbb{F}}(m)$). Let $S = \{l_1, \ldots, l_k\} \subset \mathcal{L}(T_i)$ be an LI set of linear forms. Let $D(\neq \phi) \subseteq \mathcal{L}(T_i)$. We say that (S, D) is a "*Detector Pair*" in $\mathcal{L}(T_i)$ if the following are satisfied for all $l_{k+1} \in D$:

- $\{l_1, \ldots, l_k, l_{k+1}\}$ is an LI set. Let $\mathcal{F} = fl(\{l_1, \ldots, l_k, l_{k+1}\})$. $\mathcal{F}$ is elementary in $\mathcal{L}(T_i)$ i.e. $\mathcal{F} \cap \mathcal{L}(T_i) = \{l_1, \ldots, l_k, l_{k+1}\}$. See Definition B.1.
- $\mathcal{F} \cap \mathcal{L}(T_{1-i}) \subseteq fl(\{l_1, \ldots, l_k\})$ i.e. $\mathcal{F}$ contains only those points from $\mathcal{L}(T_{1-i})$ which lie inside $fl(\{l_1, \ldots, l_k\})$.

3.4.1 Identifying Some Factors Which Don't Divide g

The two claims below give results about structure of linear forms which divide g. The proofs are easy but technical and so we move them to the appendix.

▶ **Claim 3.6.** *Let $(S = \{l_1 \dots, l_k\}, D)$ be a Detector set in $\mathcal{L}(T_i)$. Let $l_{k+1} \in D$. For a standard linear form $l \in V$, if $l \mid g$ then $l \notin sp(\{l_1, \dots, l_k\})$.*

Proof. See F.1 in appendix. ◀

▶ **Claim 3.7.** *Let $l \in Lin_{\mathbb{F}}[\bar{x}]$ be standard such that $l \mid g$ and $\mathcal{C}$ be the candidate set. Assume $(S = \{l_1, \dots, l_k\}, D(\neq \phi))$ is a Detector pair in $\mathcal{L}(T_i)$. Then $|\mathcal{L}(T_{1-i}) \cap (fl(S \cup \{l\}) \setminus fl(S))| \geq 2$. That is the flat $fl(\{l_1, \dots, l_k, l\})$ contains atleast two distinct points from $\mathcal{L}(T_{1-i})(\subseteq \mathcal{C})$ outside $fl(\{l_1, \dots, l_k\})$.*

Proof. See F.2 in appendix. ◀

▶ **Claim 3.8.** *Suppose $(S = \{l_1, \dots, l_k\}, D(\neq \phi))$ is a Detector Pair in $\mathcal{L}(T_i)$. The following algorithm identifies some factors in $\mathcal{L}(G) \setminus \mathcal{L}(g)$. It returns the product of all linear forms identified.*

FunctionName : IdentifyFactors
input : $f \in \Sigma\Pi\Sigma_{\mathbb{F}}(2)[\bar{x}], \mathcal{C} \subset Lin_{\mathbb{F}}[\bar{x}], S = \{l_1, \dots, l_k\} \subset Lin_{\mathbb{F}}[\bar{x}]$)
output : a $\Pi\Sigma_{\mathbb{F}}[\bar{x}]$ polynomial

```
1  I = 1, bool flag;
2  for each factor l of f do
3  |   flag = false;
4  |   if l, l_1, ..., l_k are LI then
5  |   |   for l'_1 ≠ l'_2 ∈ C \ fl({l_1, ..., l_k}) do
6  |   |   |   if l'_1, l'_2 ∈ sp({l, l_1, ..., l_k}) then flag = true;
7  |   |   |   break;
8  |   |   end
9  |   end
10 |   if !flag then
11 |   |   I = I×l;
12 |   end
13 end
14 return I;
```

Algorithm 3: Identify Factors.

Proof. The proof of the claim is a part of Lemma 3.9 below. ◀

3.4.1.1 Time Complexity

Since $\mathcal{C}$ has size $poly(d)$ and $deg(f) = d$, the nested loops run $poly(d)$ times. k, r are constants so checking linear independence of $k+1$ linear forms in r variables takes constant time. Checking if some vectors belong to a $k+1$ dimensional space also takes constant time. Multiplying linear forms to $\mathbf{I}$ takes $poly(d)$ time. So overall the algorithm runs in $poly(d)$ time.

So the above algorithm identified a factor $\mathbf{I}$ of G for us. Let us define new polynomials

$$G = \frac{G}{\mathbf{I}} = \prod_{t \in [N_1]} G_t$$

and

$$f = \frac{f}{\mathbf{I}} = G(\alpha_0 T_0 + \alpha_1 T_1)$$

▶ **Lemma 3.9.** *The following are true:*

1. *If $l \mid I$ (i.e. l was identified) then $l \in \mathcal{L}(G) \setminus \mathcal{L}(g)$.*
2. *If $l \mid G$ (i.e. l was retained) then $(fl(\{l_1, \ldots, l_k, l\}) \setminus fl(\{l_1, \ldots, l_k\})) \cap (\mathcal{L}(T_{1-i}) \cup (\mathcal{L}(T_i) \setminus D)) \neq \phi$ that is:*
   $(fl(\{l_1, \ldots, l_k, l\}) \setminus fl(\{l_1, \ldots, l_k\}))$ contains a point from $\mathcal{L}(T_i) \setminus D$ or $\mathcal{L}(T_{1-i})$.
3. *If $l \mid G$ and $l_{k+1} \in D$ then $l \notin sp(\{l_1, \ldots, l_k, l_{k+1}\})$.*

Proof. See F.3 in appendix. ◀

3.4.2 Overestimating the set D of the detector pair (S, D)

Lemma 3.9 is going to help us actually find an overestimate of D corresponding to $S = \{l_1, \ldots, l_k\}$ in the detector pair (S, D) as described in the lemma below. This will be important since we need D during our algorithm for the Hard Case.

▶ **Lemma 3.10.** *Let $(S = \{l_1, \ldots, l_k\}, D)$ be a detector in $\mathcal{L}(T_i)$. For each $(l, l_j) \in \mathcal{C} \times S$ define the space $U_{\{l,l_j\}} = sp(\{l, l_j\})$. Extend $\{l, l_j\}$ to a basis and in the process obtain $U'_{\{l,l_j\}}$ such that $V = U_{\{l,l_j\}} \oplus U'_{\{l,l_j\}}$. Define the set:*

$$X = \{l \in \mathcal{C} : \pi_{U'_{\{l,l_j\}}}(f) \neq 0, \text{ for all } l_j \in S\}$$

Then $D \subset X \subset \mathcal{L}(T_i)$.

Proof. See F.4 in appendix. ◀

This set X is an overestimate of D inside $\mathcal{L}(T_i)$ and also easy to compute. Given S we may easily construct X in time $poly(d)$ because of it's simple description. Let's give an algorithm to compute X given $f, S, \mathcal{C}$.

▶ **Claim 3.11.** *Algorithm 4 computes the overestimate X of D as discussed above.*

3.4.2.1 Time Complexity

Inside the inner for loop we look for $(r-2)$ linear forms from $\mathcal{C}$. $|\mathcal{C}| = poly(d)$ and r is a constant and so this step only needs $poly(d)$ time. The nested loops run polynomially many times. Checking linear independece of r linear forms and projecting to known constant dimensional subspaces also take $poly(d)$ time as has been discussed before. So the algorithm runs in $poly(d)$ time.

FunctionName : OverestimateDetector
input : $f \in \Sigma\Pi\Sigma_{\mathbb{F}}(2)[\bar{x}], S = \{l_1, \ldots, l_k\} \subset Lin_{\mathbb{F}}[\bar{x}], \mathcal{C} \subset Lin_{\mathbb{F}}[\bar{x}]$)
output : Set of linear forms

```
1  bool flag;
2  Define X ← φ;
3  for each l ∈ C do
4  |   flag = true;
5  |   for each l_j ∈ S with {l, l_j} LI do
6  |   |   Find {l'_1, ..., l'_{r-2}} ⊂ C such that {l, l_j, l'_1, ..., l'_{r-2}} is LI;
7  |   |   U ← Fl ⊕ Fl_j; U' ← Fl'_1 ⊕ ... ⊕ Fl'_{r-2};
8  |   |   if π_{U'}(f) == 0 then
9  |   |   |   flag = false;
10 |   |   |   break;
11 |   |   end
12 |   end
13 |   if flag then
14 |   |   X ← X ∪ {l};
15 |   end
16 end
17 return X;
```

Algorithm 4: Overestimate Detector.

3.5 Hard Case

$$\boxed{\mathcal{L}(T_{1-i}) \subseteq sp(U_i), \text{ for } i = 0 \text{ and } 1}$$

This subsection will involve the most non trivial ideas. We handled $dim(^{sp(T_{1-i}) + sp(T_i)}/_{sp(T_i)}) \geq 2$ in the Medium Case (see Subsection 3.3) completely, so let's assume $dim(^{sp(T_{1-i}) + sp(T_i)}/_{sp(T_i)}) \leq 1 \Rightarrow dim(\mathcal{L}(T_{1-i}) \cup \mathcal{L}(T_i)) \leq dim(\mathcal{L}(T_i)) + 1$ for both $i = 0, 1$. We already know that $rank(f) = r$, implying $dim(\mathcal{L}(T_i) \cup \mathcal{L}(T_{1-i})) = r$. Thus for $i = 0, 1$; $dim(\mathcal{L}(T_i)) \geq r - 1$. This works in our favour for applying the quantitative version of the Sylvester Gallai theorems given in [3]. To be precise we will use Lemma B.6 from Appendix B in this paper.

1. Our first application (see Lemma 3.13) of Quantitative Sylvester Gallai will help us prove the existence of a Detector pair $(S = \{l_1, \ldots, l_k\}, D)$ in $\mathcal{L}(T_i)$ with $k = c_{\mathbb{F}}(3) + 2$ (See definition of $c_{\mathbb{F}}(.)$ in Theorem 1.7) and large size of D. For this we will only need $dim(\mathcal{L}(T_i)) \geq C_{2k-1}$ for $i = 0, 1$ (see Appendix B for definition of C_{2k-1}). From Definition 1.2 we know that this is true with $k = c_{\mathbb{F}}(3) + 2$.
2. The above point shows the existence of a detector pair (S, D) in $\mathcal{L}(T_i)$ with large $|D|$. So now we go back to Subsection 3.4 and remove some factors of f to get $f = G(\alpha_0 T_0 + \alpha_1 T_1)$ such that linear factors of G satisfy properties given in Lemma 3.9. We also compute the overestimate X of D using Algorithm 4. Let the known and unknown parts of f be $K_0^\star, K_1^\star$ and $U_0^\star, U_1^\star$ respectively. If for some $i \in \{0, 1\}$, $\mathcal{L}(T_i) \subsetneq sp(U_{1-i})$ then we are in Easy Case for f and can recover the gates for f. Otherwise for both $i = 0, 1$; $\mathcal{L}(T_i) \subseteq sp(U_{1-i}) \Rightarrow dim(\mathcal{L}(U_{1-i})) \geq r - 1$ and we continue with reconstruction below.
3. Next to actually reconstruct linear forms in U_{1-i}, we will use it's high-dimensionality ($\geq r - 1 \geq C_{2k-1}$) discussed above. Lemma B.6 from Appendix B will enable us to

prove the existence of a $d_1 \in D$ which together with the set S found above will give the existence of a "*Reconstructor*" (see Claim C.4 and Algorithm 7) which recovers some linear factors of U_{1-i} with multiplicity (see Theorem 3.14).

3.5.1 Large Size of Detector Sets

W.l.o.g. we assume $|\mathcal{L}(T_0)| \leq |\mathcal{L}(T_1)|$. First we point out a simple calculation that will be needed later. For $\delta \in (0, \frac{7-\sqrt{37}}{6})$ and $\theta \in (\frac{3\delta}{1-\delta}, 1-3\delta)$, let $v(\delta,\theta)$ be defined as follows:

$$v(\delta,\theta) = \begin{cases} 1-\delta-\theta & \text{if } |\mathcal{L}(T_0)| \leq \theta|\mathcal{L}(T_1)| \\ (1-\delta)(1+\theta)-1 & \text{if } \theta|\mathcal{L}(T_1)| < |\mathcal{L}(T_0)| \leq |\mathcal{L}(T_1)| \end{cases}$$

▶ **Claim 3.12.** *The following is true*

$$\frac{(2-v(\delta,\theta))}{v(\delta,\theta)} \leq \frac{1-\delta}{\delta}\,.$$

Proof. See G.1 in appendix. ◀

▶ **Lemma 3.13.** *Let $k = c_{\mathbb{F}}(3)+2$ (see defn of $c_{\mathbb{F}}(m)$ in Theorem 1.7). Fix δ, θ in range given in Claim 3.12 above. Then for some $i \in \{0,1\}$ there exists a Detector $(S = \{l_1, \ldots, l_k\}, D)$ in $\mathcal{L}(T_i)$ with $|D| \geq v(\delta,\theta)\max(|\mathcal{L}(T_0)|, |\mathcal{L}(T_1)|)$.*

Proof. See G.2 in appendix. ◀

3.5.2 Assuming $\mathcal{L}(T_i) \subseteq sp(\mathcal{L}(U_{1-i}))$ and reconstructing factors of U_{1-i}

Let's begin by stating our main reconstruction theorem for this Subsubsection. We will go through several steps to prove it:

▶ **Theorem 3.14.** *There exist pairwise disjoint LI sets S_0, S_1, S_2 with $S_0 \cup S_1 \cup S_2$ being a basis of $V = Lin_{\mathbb{F}}[x_1, \ldots, x_r] \simeq \mathbb{F}^r$, and non constant polynomials P, Q dividing U_{1-i} such that $P \mid Q$ and (Q, P, S_0, S_1, S_2) is a Reconstructor.*

Once we know this result we actually recover P by computing $\pi_{W_0'}(Q)$ and $\pi_{W_1'}(Q)$ and then using Algorithm 7. We state this in the following corollary. Proof is given as Algorithm 5

▶ **Corollary 3.15.** *Using $f, K_{1-i}, S_0, S_1, S_2$ from above we can compute $\pi_{W_0'}(Q), \pi_{W_1'}(Q)$ for Q defined in the proof above.*

Before going to the proof let's do some more more preparation.

Consider the set of linear forms $\mathcal{X} = \mathcal{L}(G) \cup \mathcal{L}(T_0) \cup \mathcal{L}(T_1)$. We know that $sp(\mathcal{X}) = V = Lin_{\mathbb{F}}[\bar{x}] \simeq \mathbb{F}^r$ (By abuse of notation we will use linear forms as points in $\mathbb{F}^r$ wherever required). Let $(S_0 = \{l_1, \ldots, l_k\}, D)$ be a detector in $\mathcal{L}(T_i)$ with $|D| \geq v(\delta,\theta)\max(|\mathcal{L}(T_0)|, |\mathcal{L}(T_1)|)$ as obtained in the preceeding discussion.

Define $W_0 = sp(S_0)$ and extend S_0 to a basis $\{l_1, \ldots, l_k, l'_{k+1}, \ldots, l'_r\}$. Now it's time to use the other random matrix Λ. Since we had applied Ω in the beginning, $\{\Omega^{-1}(l_1), \ldots, \Omega^{-1}(l_k)\}$ are linear forms in our input polynomial for this section. By Assumption 3 we know that the set

$$\{\Omega(\Omega^{-1}l_1), \ldots, \Omega(\Omega^{-1}l_k), \Lambda\Omega(\Omega^{-1}l'_{k+1}), \ldots, \Lambda\Omega(\Omega^{-1}l'_r)\}$$

is LI. Let $l_j = \Lambda l'_j, j \in \{k+1, \ldots, r\}$. So $\mathcal{B} = \{l_1, \ldots, l_r\}$ is a basis. and define $W_0^\perp = sp(\{l_{k+1}, \ldots, l_r\})$. Clearly $V = W_0 \oplus W_0^\perp$.

By Assumption 4 for any $l \in \mathcal{X} \setminus W_0$, $[l]_{\mathcal{B}}^{k+1} \neq 0$. We re-normalize all linear forms in $\mathcal{X} \setminus W_0$ making sure that the coefficient of l_{k+1} is 1 in them. From now onwards this will be assumed.

With this notation we proceed towards detecting linear factors of the unknown parts. But first let's show that even after projecting onto $W_0^\perp$, the detector is larger in size (upto a function of δ) compared to one of the unknown parts.

▶ **Lemma 3.16.** *The following are true:*
1. $dim(\pi_{W_0^\perp}(\mathcal{L}(U_{1-i}))) > C_4$
2. $\pi_{W_0^\perp}(\mathcal{L}(U_{1-i})) \cap \pi_{W_0^\perp}(D) = \phi$
3. $|\pi_{W_0^\perp}(\mathcal{L}(U_{1-i})) \setminus \{0\}| \leq \frac{1-\delta}{\delta} |\pi_{W_0^\perp}(D)|$

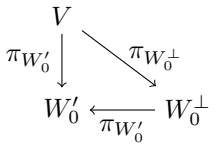
Proof. See G.3 in appendix. ◀

This lemma enables us to apply Lemma B.6 from Appendix B. Consider the sets $Y = \pi_{W_0^\perp}(\mathcal{L}(U_{1-i})) \setminus \{0\}$ and $X = \pi_{W_0^\perp}(D)$. We've shown all conditions in Lemma 2.5, so there exists a line $\vec{L}$ (called a "*semiordinary bichromatic*" line) in $W_0^\perp$ such that $|\vec{L} \cap Y| = 1$ and $|\vec{L} \cap X| \geq 1$.

Let's prove another short lemma which is useful for technical reasons.

▶ **Lemma 3.17.** *For any subspace W'_0 such that $V = W_0 \oplus W'_0 = W_0 \oplus W_0^\perp$ there is a line $\vec{L} \subset W'_0$ such that*
1. $|\vec{L} \cap \pi_{W'_0}(D)| \geq 1$
2. $|\vec{L} \cap (\pi_{W'_0}(\mathcal{L}(U_{1-i})) \setminus \{0\})| = 1$

Proof. We have the following commutative diagram:



Let $v = w + w^\perp \in V$ where $w \in W_0, w^\perp \in W_0^\perp$, then

$$\pi_{W'_0}(\pi_{W_0^\perp}(v)) = \pi_{W'_0}(w^\perp) = \pi_{W'_0}(w^\perp) + \pi_{W'_0}(w) = \pi_{W'_0}(v)$$

So $\pi_{W'_0} = \pi_{W'_0} \circ \pi_{W_0^\perp}$

Next let $T : V \to V$ be any bijection then $T(A \cap B) = T(A) \cap T(B)$ and therefore $|A \cap B| = |T(A) \cap T(B)|$. Since the maps above are projections one can easily see that $\pi_{W'_0} : W_0^\perp \to W'_0$ is an isomorphism where the inverse of any $w' \in W'_0$ is given as $\pi_{W_0^\perp}(w')$. Call this map T. Now any linear isomorphism between vector spaces also preserves affine dependence since:

$$T(\lambda u + (1-\lambda)v) = \lambda T(u) + (1-\lambda)T(v)\,.$$

So image of a line is a line. Let $\vec{L}$ be the line obtained in Lemma 3.16.

- $T(\vec{L})$ is a line in W'_0.

- $|T(\vec{L}) \cap \pi_{W_0'}(D)| = |T(\vec{L}) \cap T(\pi_{W_0^\perp}(D))| = |\vec{L} \cap \pi_{W_0^\perp}(D)| \geq 1$
- $|T(\vec{L}) \cap \pi_{W_0'}(\mathcal{L}(U_{1-i}))| = |T(\vec{L}) \cap T(\pi_{W_0^\perp}(\mathcal{L}(U_{1-i})))| = |\vec{L} \cap \pi_{W_0^\perp}(\mathcal{L}(U_{1-i}))|$

Since T is a linear isomorphism $0 \in \pi_{W_0^\perp}(\mathcal{L}(U_{1-i})) \Leftrightarrow 0 \in T(\pi_{W_0^\perp}(\mathcal{L}(U_{1-i}))) = \pi_{W_0'}(\mathcal{L}(U_{1-i}))$ and $0 \in \vec{L} \Leftrightarrow 0 \in T(\vec{L})$, therefore the third condition above is same as

$$|T(\vec{L}) \cap (\pi_{W_0'}(\mathcal{L}(U_{1-i})) \setminus \{0\})| = |\vec{L} \cap (\pi_{W_0^\perp}(\mathcal{L}(U_{1-i})) \setminus \{0\})| = 1$$

So the lemma is true with $\vec{L}$ being the line $T(\vec{L})$ obtained in this proof. ◀

Finally it's time to give the proof of Theorem 3.14. Let $d_1 \in D$ such that $\pi_{W_0^\perp}(d_1) \in \vec{L}$ where $\vec{L}$ was the line obtained rightafter Lemma 3.16. Since coefficient of l_{k+1} is non-zero in d_1, $\{l_1, \ldots, l_k, d_1, l_{k+2}, \ldots, l_r\}$ is also a basis. Define $S_0 = \{l_1, \ldots, l_k\}, S_1 = \{d_1\}, S_2 = \{l_{k+2}, \ldots, l_r\}, W_i = sp(S_i), W_i' = \bigoplus_{j \neq i} W_j$. Note this implies $V = W_0 \oplus W_0'$ and so Lemma 3.17 above can be used. Let $\vec{L}$ be the line the Lemma 3.17 gives. *By re-normalization we also assume that all linear forms in $\mathcal{X} \setminus W_0'$ have coefficient of d_1 equal to* 1.

Proof of Theorem 3.14

We show this in steps:

- Let S_0, S_1, S_2 be as defined in the discussion above.
- Let Q be the largest factor of U_{1-i} such that for all linear forms $q \mid Q$, $\pi_{W_2}(q) \neq 0$. So $\pi_{W_2}(Q) \neq 0$ and if $u^\star \mid \frac{U_{1-i}}{Q}$ is a linear form then $\pi_{W_2}(u^\star) = 0$. Let P be the $\Pi\Sigma$ polynomial with the largest possible degree such that for all linear factors p of P, $\pi_{W_0'}(p) \in \vec{L} \cap (\pi_{W_0'}(\mathcal{L}(U_{1-i})) \setminus \{0\})$. Clearly P is non constant since $|\vec{L} \cap (\pi_{W_0'}(\mathcal{L}(U_{1-i})) \setminus \{0\})| = 1$. Clearly $\pi_{W_0'}(P) \neq 0 \Rightarrow P \mid Q$. Then (Q, P, S_0, S_1, S_2) is a *Reconstructor* (See Subsection C for definition) for P. Let's check this is true:
  - $\pi_{W_2}(Q) \neq 0$ - By definition of Q we know this for all it's factors and therefore for Q itself.
  - $\pi_{W_0'}(P) = \pi_{W_0'}(p)^t$, for some linear form $p \mid P$ (since $|\vec{L} \cap (\pi_{W_0'}(\mathcal{L}(U_{1-i})) \setminus \{0\})| = 1$).
  - Let $q \mid \frac{Q}{P}$ such that $gcd(\pi_{W_2}(P), \pi_{W_2}(q)) \neq 1 \Rightarrow$ there exists some linear factor $p \mid P$ such that $\pi_{W_2}(p), \pi_{W_2}(q)$ are LD. $\{\pi_{W_2}(p), \pi_{W_2}(q)\}$ are LD and non-zero implies $q \in sp(\{l_1, \ldots, l_k, d_1, p\}) \Rightarrow \pi_{W_0'}(q) \in sp(\{\pi_{W_0'}(d_1), \pi_{W_0'}(p)\}) = sp(\{d_1, \pi_{W_0'}(p)\})$. So clearly $\pi_{W_0'}(q) \in sp(\{d_1, \pi_{W_0'}(p)\})$. Since coefficient of d_1 in $\pi_{W_0'}(q), d_1$, and $\pi_{W_0'}(p)$ is 1, and therefore using Lemma 1.10 it's easy to see that $\pi_{W_0'}(q) \in fl(\{d_1, \pi_{W_0'}(p)\}) = \vec{L}$. Since $Q \mid U_{1-i}$ we have $\pi_{W_0'}(q) \in \pi_{W_0'}(\mathcal{L}(U_{1-i})) \setminus \{0\} \Rightarrow \pi_{W_0'}(q) \in \vec{L} \cap (\pi_{W_0'}(\mathcal{L}(U_{1-i})) \setminus \{0\}) = \{\pi_{W_0'}(p)\}$ which can't be true since P is the largest polynomial dividing Q where linear factors have this property and $q \nmid P$. So such a q does not exist.

Now we give the algorithm for reconstruction in this case, see Algorithm 5.

3.5.2.1 Correctness

Let's assume we returned an object *obj* of type decomposition.

1. **If** $obj \to iscorrect == true$**:** then we ought to be right since we check if $obj \to f = obj \to M_0 + obj \to M_1$. Since the representation is unique this will be the correct answer.
2. **If** $obj \to iscorrect == false$**:** Let's assume f actually has a $\Sigma\Pi\Sigma(2)$ representation. If we were in Easy Case or Medium Case we would have already found the circuit using their algorithms. So we are in the Hard Case. So by Lemma 3.13 there exists i such that

FunctionName : HardCase
Fix : $k = c_{\mathbb{F}}(3) + 2$
input : $f \in \Sigma\Pi\Sigma_{\mathbb{F}}(2)[\bar{x}], \mathcal{C} \subset Lin_{\mathbb{F}}[\bar{x}], \Lambda \in \mathbb{F}^{r\times r}$
output : An object of type *decomposition*

for $i \leftarrow 0$ **to** 1 **do**
  for *each LI* $\mathcal{B}' = \{l_1, \ldots, l_k, l'_{k+1}, \ldots, l'_r\} \subset \mathcal{C}$ **do**
    $S_0 = \{l_1, \ldots, l_k\}$;
    for $j \leftarrow k+1$ **to** r **do**
      $l_j \leftarrow \Lambda(l'_j)$;
    end
    if $\mathcal{B} = \{l_1, \ldots, l_r\}$ *is LI* **then**
      $I \leftarrow IdentifyFactors(f, \mathcal{C}, S_0)$;
      if $I \mid f$ **then**
        $f \leftarrow \frac{f}{I}$, $K_0^\star = 1, K_1^\star = 1$, $X \leftarrow$ OverestDetector$(f^\star, \mathcal{C}, S_0)$;
        while $deg(K_{1-i}^\star) < deg(f)$ **do**
          if $EasyCase(f, K_0^\star, K_1^\star, \mathcal{C}) \rightarrow iscorrect$ **then**
            return object $decomposition(f, IK_0^\star, IK_1^\star)$;
          end
          else
            for *each* $d_1 \in X$ **do**
              if $\mathcal{B}_2 = \{l_1, \ldots, l_k, d_1, l_{k+2}, \ldots, l_r\}$ *is LI* **then**
                $V_j = \mathbb{F}l_j, j \in [r] \setminus \{k+1\}$, $V_{k+1} = \mathbb{F}d_1$, $V'_j = \bigoplus_{t\in[r]\setminus\{j\}} V_t$;
                $S_0 = \{l_1, \ldots, l_k\}, S_1 = \{u_{k+1}\}, S_2 = \{l_{k+2}, \ldots, l_r\}$;
                $W_j = sp(S_j), W'_j = \bigoplus_{j_1 \neq j} W_{j_1}$ for $j \in \{0, 1, 2\}$;
                $Q_0 = \frac{\pi_{V'_1}(f)}{\pi_{V'_1}(K_{1-i}^\star)}$, $Q_1 = \frac{\pi_{W'_1}(f)}{\pi_{W'_1}(K_{1-i}^\star)}$;
                if $Q_0, Q_1 \in \Pi\Sigma[\bar{x}]$ *and non-zero* **then**
                  for $q_0 \mid Q_0$ *&&* $q_0 \in W'_2$, $q_1 \mid Q_1$ *&&* $q_1 \in W'_2$ **do**
                    $Q_0 = \frac{Q_0}{q_0}, Q_1 = \frac{Q_1}{q_1}$;
                  end
                  $Q_0 = \pi_{W'_0}(Q_0)$;
                  if $deg(Reconstructor(Q_0, Q_1, S_0, S_1, S_2)) \geq 1$ **then**
                    $K_{1-i}^\star \leftarrow K_{1-i}^\star \times Reconstructor(Q_0, Q_1, S_0, S_1, S_2)$;
                  end
                end
              end
            end
          end
        end
        if $f - IK_{1-i}^\star \in \Pi\Sigma[\bar{x}]$ **then**
          $M_0 = IK_{1-i}^\star$, $M_1 = f - M_0$, **return** *new object* $decomposition(f, M_0, M_1)$;
        end
      end
    end
  end
end
return $decomposition()$;

Algorithm 5: Hard Case Reconstruction.

$\mathcal{L}(T_i)$ has a detector pair (S_0, D) with $|D|$ large. For this i there exists such an S_0, so sometime during the algorithm we would have guessed the correct i and the correct S_0. *Now let's analyze what happens inside the while and the third for loop when the first two guesses are correct.* Note that this also implies that the I we have identified is correct and now we need to solve for

$$f = G(\alpha_0 T_0 + \alpha_1 T_1)$$

Let $K_0^\star, K_1^\star$ (initialized to 1) be the known parts of the gates for this polynomial f and $U_0^\star, U_1^\star$ be the unknown parts. Note that T_0, T_1 are same for both polynomials so $rank(f) = rank(f)$ and for $j = 0, 1$; $K_j \mid GT_j$.

Assume until the m^{th} iteration of the while loop $K_t^\star \mid GT_t$ for $t \in \{0,1\}$, we show that after the $(m+1)^{th}$ iteration, this property continues to hold and $deg(K_{1-i}^\star)$ increases.

- If after the m^{th} iteration of the while loop for some $j \in \{0,1\}$, $\mathcal{L}(T_j) \subsetneq sp(\mathcal{L}(U_{1-j}^\star))$ we are in Easy Case for f. The first step in while loop is to call EasyCase$(f, \mathcal{C}, K_0^\star, K_1^\star)$. This will clearly recover the circuit for f and return true since $K_t^\star \mid GT_t$ for $t \in \{0,1\}$. However this does not happen so for both $j = 0, 1$, we have $\mathcal{L}(T_i) \subsetneq \mathcal{L}(U_{1-i})$. This means that we can use the ideas in Subsection 3.5.2, specifically Theorem 3.14.
- The first two guesses are correct imply that $D \subseteq X \subseteq \mathcal{L}(T_i)$.
- If d gets rejected then $K_t, t \in \{0,1\}$ remain unchanged. If some d_1 does not get rejected then since $d_1 \in \mathcal{L}(T_i)$, $Q_0 = \pi_{V_1'}(U_{1-i})$ is a non zero $\Pi\Sigma$ polynomial. Then some factors (the ones $\in W_2'$) are removed from Q_0. Also on projecting to W_0' this still remains non-zero (as d_1 was not rejected).
- We know that $d_1 \in \mathcal{L}(T_i)$ and d_1 not getting rejected implies that $Q_1 = \pi_{W_1'}(U_{1-i})$ is a non-zero $\Pi\Sigma$ polynomial. We again remove some factors (i.e. the ones in W_2') from Q_1. The non-zeroness of Q_0, Q_1 imply that $Q_0 = \pi_{W_1'}(Q), Q_1 = \pi_{W_1'}(Q)$ i.e. they are projections of the same polynomial Q which is the largest factor of U_{1-i} with the property that any linear form $q \mid Q$ is not in $W_2' = W_0 \oplus W_1$.
- d_1 was not rejected implies that $Reconstructor(Q_0, Q_1, S_0, S_1, S_2)$ returned a non-trivial polynomial P. This has to be a factor of Q by Claim C.6 following Algorithm 7 and therefore a factor of U_{1-i}.
- Proof of Theorem 3.14 implies that in every iteration atleast some d_1 will not be rejected.
- So clearly the new $K_{1-i}^\star = K_{1-i}^\star \times P$ divides GT_{1-i}. K_i remains unchanged. Therefore even after the $(m+1)^{th}$ iteration $K_t \mid GT_t$ for both $j = 0, 1$ but degree of $K_{1-i}^\star$ increases.

So the while loop cannot run more than $deg(f)$ times and in the end GT_{1-i} will be reconstructed completely and correctly and we should have returned obj with $obj \to iscorrect = true$. Therefore we have a contradiction and so f did not have a $\Sigma\Pi\Sigma(2)$ circuit and we correctly returned false.

Running Time

- First for loop runs twice.
- Inside it chossing r linear forms from $\mathcal{C}$ ($|\mathcal{C}| = poly(d)$) takes $poly(d)$ time.
- Applying Λ to $r - k$ vectors takes $poly(r) = O(1)$ time.

- Checking if a set of size r inside $\mathbb{F}^r$ is LI takes $poly(r) = O(1)$ time since it is equivalent to computing determinant.
- $IdentifyFactors()$ takes $poly(d)$ time and computing f also takes $poly(d)$ time.
- $OverestDetector()$ runs in $poly(d)$ time.
- while loop runs atmost d times
- $EasyCase$ runs in $poly(d)$ time and so does polynomial multiplication.
- $X \subseteq \mathcal{L}(T_i)$ and $|\mathcal{L}(T_i)| \leq deg(f)$ and so for loop runs d times.
- Change of bases in $\mathbb{F}^r$ and application to a polynomial of degree d takes $poly(d)$ time.
- Therefore projecting to subspaces also takes $poly(d)$ time.
- $Reconstructor()$ runs in $poly(d)$ time (since r is a constant) and so does polynomial multiplication and factoring by [14].

Since all of the above steps run in $poly(d)$ time, nesting them a constant number of times also takes $poly(d)$ time. Therefore the running time of our algorithm is $poly(d)$.

3.6 Algorithm including all cases

The algorithm we give here will be the final algorithm for rank r $\Sigma\Pi\Sigma$ polynomials. It will use the previous three cases. Our input will be a $\Sigma\Pi\Sigma(2)$ polynomial $f(x_1, \ldots, x_r)$ and output will be a circuit computing the same.

FunctionName : RECONSTRUCT
input : $f \in \Sigma\Pi\Sigma_{\mathbb{F}}(2)[\bar{x}]$
output : An object of type *decomposition*

decomposition obj;
$(\Omega_{i,j}), (\Lambda_{i,j})$, $r \times r$ matrices with entries chosen uniformly randomly from $[N]$;
$L_i(\bar{x}) \leftarrow \sum_{k=1}^{r} \Omega_{i,k} x_k$;
$f(x_1, \ldots, x_r) \leftarrow f(L_1(\bar{x}), \ldots, L_r(\bar{x}))$;
$\mathcal{C} \leftarrow Candidates(f(x_1, \ldots, x_r))$;
if $MediumCase(f, \mathcal{C})) \rightarrow iscorrect$ **then**
  obj $\leftarrow$ MediumCase$(f, \mathcal{C})$;
end
else if $EasyCase(f, K_0, K_1, \mathcal{C}) \rightarrow iscorrect$ **then**
  obj $\leftarrow$ EasyCase$(f, K_0, K_1, \mathcal{C})$;
end
else
  obj $\leftarrow$ HardCase$(f, \mathcal{C}, \Lambda)$;
end
Apply Ω^{-1} to obj$\rightarrow f$, obj$\rightarrow M_0$, obj$\rightarrow M_1$;
return *obj;*

Algorithm 6: Reconstruction in low rank.

Explanation

Here we explain every step of the given algorithm:

- The function RECONSTRUCT(f) takes as input a polynomial $f \in \Sigma\Pi\Sigma_{\mathbb{F}}(2)[\bar{x}]$ of $rank = r$ and outputs two polynomials $K_0, K_1 \in \Pi\Sigma_{\mathbb{F}}[\bar{x}]$ which are the two gates in it's circuit representation.

- Steps 2, 3 picks a random matrix Ω and transforms each variable using the linear transformation this matrix defines. With high probability this will be an invertible transformation. We do the reconstruction for this new polynomial since the linear factors of it's gates satisfy some non-degenerate conditions(because they have been randomly transformed) our algorithm needs. We apply Ω^{-1} after the reconstruction and get back our original f.
- The next step constructs the set of candidate linear forms $\mathcal{C}$. We've talked about the size, construction and structure of this set in Section E.
- We first assume Medium Case. It that was not the case we check for Easy Case. If both did not occur we can be sure we are in the Hard case.
- We apply Ω^{-1} to polynomials in obj and return it.

4 Reconstruction for arbitrary *rank*

This section reduces the problem from $\Sigma\Pi\Sigma(2)$ Circuits with arbitrary rank n ($> s$) to one with constant rank r (still $> s$). Also once the problem has been solved efficiently in the low rank case we use multiple instances of such solutions to lift to the general $\Sigma\Pi\Sigma(2)$ circuit. The idea is to project the polynomial to a small (polynomial) number of random subspaces of dimension r, reconstruct these low rank polynomials and then lift back to the original polynomial. The uniqueness of our circuit's representation plays a major role in both the projection and lifting steps. Let

$$f = G(\alpha_0 T_0 + \alpha_1 T_1)$$

G, T_i are normal $\Pi\Sigma$ polynomials. All notations are borrowed from the previous section. It is almost identical to the restriction done in [24] except that the dimension of random subspaces is different. For more details see Section 4.2.1 and 4.2.3. in [24]. Since all proofs have been done in detail in [24] we do not spend much time here. A clear sketch with some proofs is however given.

4.1 Projection to a Random Low Dimensional Subspace

We explain the procedure of projecting to the random subspace below. In this low dimensional setup we can get coefficient representation of $\pi_V(f)$, also some important properties of f are retained by $\pi_V(f)$. Proofs are simple and standard so we discuss them in the appendix at end.

Pick n vectors $v_i, i \in [n]$ with each co-ordinate chosen independently from the uniform distribution on $[N]$. Let $V = sp(\{v_i : i \in [r]\})$ and $V' = sp\{v_i : i \in \{r+1, \ldots, n\}\}$. Then $V \oplus V' = \mathbb{F}^n$ Let π_V denote the orthogonal projection onto V. With high probability the following hold:

1. $\{v_i : i \in [n]\}$ is linearly independent (see Appendix H.1 for proof).
2. Let $\{l_1, \ldots, l_r\}$ be a set of r linearly independent linear forms in $\mathcal{L}(T_0) \cup \mathcal{L}(T_1)$. Then $\pi_V(\{l_1, \ldots, l_r\})$ is linearly independent with high probability. So $rank(\pi_V(f)) = r$ (see Appendix H.2 for Proof).
3. Let $l_{01} \in \mathcal{L}(T_0), l_{11} \in \mathcal{L}(T_1)$, then $\pi_V(l_{01}), \pi_V(l_{11})$ are linearly independent with high probability and so $gcd(\pi_V(T_0), \pi_V(T_1)) = 1$.

Pick large number of ($\geq d^r$) random points $p_i, i = 1, \ldots, d^r$ in the space V. Use the values $\{f(p_i)\}$ and get a coefficient representation for $\pi_V(f)$. With high probability over the choice of points interpolation will work (See Appendix H.3 for Proof). We will effectively be solving a linear system. Note that the number of coefficients in $f|_V = O(d^r)$. Now this coefficient

representation of $\pi_V(f)$ is reconstructed using the algorithm in Section 3. A number of such reconstructions are then glued to reconstruct the original polynomial.

4.2 Lifting from the Random Low Dimensional Subspace

1. Consider spaces $V_i = V \oplus \mathbb{F}v_i$ for $i = r+1, \ldots, n$.
2. Reconstruct $\pi_{V_i}(f)$ and $\pi_V(f)$ for each $i \in \{r+1, \ldots, n\}$.
3. Let $l = \sum_{i=1}^{n} a_i v_i$ be a linear form dividing one of the gates of f say T_0. $\pi_V(l) = \sum_{i=1}^{r} a_i v_i$ and $\pi_{V_i}(l) = \sum_{j=1}^{r} a_j v_j + a_i v_i$. Using our algorithm discussed in Section 3 we would have reconstructed $\pi_V(f)$ and $\pi_{V_i}(f)$. So we know the triples $(\pi_V(G), \pi_V(T_0), \pi_V(T_1))$ and $(\pi_{V_i}(G), \pi_{V_i}(T_0), \pi_{V_i}(T_1))$
   On restricting V_i to V:
   a. *Only Factors become factors* with high probability so we can easily find the correspondence between $\pi_V(G)$ and $\pi_{V_i}(G)$.
   b. $\pi_V(\pi_{V_i}(T_0)) = \pi_V(T_0)$ and $\neq \pi_V(T_1)$ because of uniqueness of representation and therefore we get the correspondence between gates.
   c. Now to get correspondence between linear forms. Let $\pi_V(l)$ have multiplicity k in $\pi_V(T_0)$. Then with high probability l has multiplicity k in T_0 Since two LI vectors remain LI on projecting to a random subspace of dimension ≥ 2 (again see Appendix H.2 for proof). Therefore $\pi_{V_i}(l)$ has multiplicity k and is the unique lift of $\pi_V(l)$ for all i. Let $\pi_{V_i}(l) = \pi_V(l) + a_i v_i$. Then $l = \pi_V(l) + \sum_{i=r+1}^{n} a_i v_j$. This finds G, T_0, T_1 for us

4.3 Time Complexity

- Interpolation to find coefficient representation $\pi_V(f)$ which is a degree d polynomial over r variables clearly takes $poly(d^r)$ time (accounts to solving a linear system of size $poly(d^r)$).
- Solving $n-r$ instances of the low rank problem (simple ranks r and $r+1$) takes $npoly(d^r)$ time.
- The above mentioned approach to glue the linear forms in the gates clearly takes $poly(n, d)$ time.
- Overall the algorithm takes $poly(n, d)$ time since r is a constant.

5 Conclusion and Future Work

We described an efficient randomized algorithm to reconstruct circuit representation of multivariate polynomials which exhibit a $\Sigma\Pi\Sigma(2)$ representation. Our algorithm works for all polynomials with rank(number of independent variables greater than a constant r). In future we would like to address the following:

- **Reconstruction for Lower Ranks:** As can be seen in the paper, rank of the polynomial for uniqueness (i.e. $c_{\mathbb{F}}(4)$) and the rank we've assumed in the low rank reconstruction (i.e. r) are both $O(1)$ but $c_{\mathbb{F}}(4)$ is smaller than r. Since one would expect a reconstruction algorithm whenever the circuit is unique we would like to close this gap.
- $\Sigma\Pi\Sigma(k)$ **circuits:** The obvious next step would be to consider more general top fan-in. In particular we could consider $\Sigma\Pi\Sigma(k)$ circuits with $k = O(1)$.
- **Derandomization:** We would like to derandomize the algorithm as it was done in the finite field case in [15].

Acknowledgements. I am extremely thankful to Neeraj Kayal for introducing me to this problem. Sukhada Fadnavis, Neeraj Kayal and myself started working on the problem together during my summer internship at Microsoft Research India Labs in 2011. We solved the first important case together. I'm grateful to them for all helpful discussions, constant guidance and encouragement.

I would like to thank Zeev Dvir for communicating the most recent rank bounds on $\delta - SG_k$ configurations from [6] and his feedback on the work. This reduces the gap in the first problem we mentioned above.

I would also like to thank Vinamra Agrawal, Pravesh Kothari and Piyush Srivastava for helpful discussions. I would also like to thank the anonymous reviewers for their suggestions. Lastly I would like to thank Microsoft Research for giving me the opportunity to intern at their Bangalore Labs with the Applied Mathematics Group.

References

1 Manindra Agrawal. Proving lower bounds via pseudo-random generators. In *FSTTCS 2005: Foundations of Software Technology and Theoretical Computer Science, 25th International Conference, Hyderabad, India, December 15-18, 2005, Proceedings, volume 3821 of Lecture*, pages 92–105. Springer, 2005.

2 V. Arvind, Partha Mukhopadhyay, and Srikanth Srinivasan. New results on noncommutative and commutative polynomial identity testing. *2012 IEEE 27th Conference on Computational Complexity*, 0:268–279, 2008. doi:10.1109/CCC.2008.22.

3 B. Barak, Z. Dvir, A. Wigderson, and A. Yehudayoff. Rank bounds for design matrices with applications to combinatorial geometry and locally correctable codes. In *Proceedings of the 43rd annual ACM symposium on Theory of computing*, STOC'11, pages 519–528, New York, NY, USA, 2011. ACM.

4 Amos Beimel, Francesco Bergadano, Nader H. Bshouty, Eyal Kushilevitz, and Stefano Varricchio. Learning functions represented as multiplicity automata. *J. ACM*, 47(3):506–530, May 2000. doi:10.1145/337244.337257.

5 B. Buchberger. A theoretical basis for the reduction of polynomials to canonical forms. *SIGSAM Bull.*, 10(3):19–29, August 1976. doi:10.1145/1088216.1088219.

6 Z. Dvir, S. Saraf, and A. Wigderson. Improved rank bounds for design matrices and a new proof of Kelly's theorem. Forum of mathematics – Sigma (to appear), 2012.

7 Zeev Dvir and Amir Shpilka. Locally decodable codes with 2 queries and polynomial identity testing for depth 3 circuits. *SIAM J. COMPUT*, 36(5):1404–1434, 2007.

8 Izrail Moiseevitch Gelfand, Mikhail M. Kapranov, and Andrei V. Zelevinsky. *Discriminants, resultants, and multidimensional determinants*. Mathematics: theory & applications. Birkhäuser, Boston, Basel, Berlin, 1994. Autre tirage de l'édition Birkhäuser chez Springer Science+ Business Media.

9 Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *J. ACM*, 33(4):792–807, August 1986. doi:10.1145/6490.6503.

10 Ankit Gupta, Neeraj Kayal, and Satya Lokam. Reconstruction of depth-4 multilinear circuits with top fan-in 2. In *Proceedings of the Forty-fourth Annual ACM Symposium on Theory of Computing*, STOC'12, pages 625–642, New York, NY, USA, 2012. ACM. doi:10.1145/2213977.2214035.

11 Ankit Gupta, Neeraj Kayal, and Satyanarayana V. Lokam. Efficient reconstruction of random multilinear formulas. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, pages 778–787, 2011. doi:10.1109/FOCS.2011.70.

12 Ankit Gupta, Neeraj Kayal, and Youming Qiao. Random arithmetic formulas can be reconstructed efficiently. *computational complexity*, 23(2):207–303, 2014. doi:10.1007/s00037-014-0085-0.

13 J. Heintz and C. P. Schnorr. Testing polynomials which are easy to compute (extended abstract). In *Proceedings of the Twelfth Annual ACM Symposium on Theory of Computing*, STOC'80, pages 262–272, New York, NY, USA, 1980. ACM. doi:10.1145/800141.804674.

14 Erich Kaltofen and Barry M. Trager. Computing with polynomials given byblack boxes for their evaluations: Greatest common divisors, factorization, separation of numerators and denominators. *J. Symb. Comput.*, 9(3):301–320, March 1990. doi:10.1016/S0747-7171(08)80015-6.

15 Zohar S. Karnin and Amir Shpilka. Reconstruction of generalized depth-3 arithmetic circuits with bounded top fan-in. In *Proceedings of the 24rd Annual CCC*, pages 274–285, 2009.

16 Neeraj Kayal and Shubhangi Saraf. Blackbox polynomial identity testing for depth 3 circuits. In *Proceedings of the 2009 50th Annual IEEE Symposium on Foundations of Computer Science*, FOCS'09, pages 198–207, Washington, DC, USA, 2009. IEEE Computer Society. doi:10.1109/FOCS.2009.67.

17 Michael Kearns and Leslie Valiant. Cryptographic limitations on learning boolean formulae and finite automata. *J. ACM*, 41(1):67–95, January 1994. doi:10.1145/174644.174647.

18 Michael Kharitonov. Cryptographic lower bounds for learnability of boolean functions on the uniform distribution. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT'92, pages 29–36, New York, NY, USA, 1992. ACM. doi:10.1145/130385.130388.

19 Adam Klivans and Amir Shpilka. Learning restricted models of arithmetic circuits. *Theory of computing*, 2(10):185–206, 2006.

20 Adam R. Klivans and Daniel Spielman. Randomness efficient identity testing of multivariate polynomials. In *Proceedings of the Thirty-third Annual ACM Symposium on Theory of Computing*, STOC'01, pages 216–223, New York, NY, USA, 2001. ACM. doi:10.1145/380752.380801.

21 Nitin Saxena. Progress on polynomial identity testing. *Bulletin of the EATCS*, 2009(99):49–79, 2009.

22 Nitin Saxena and C. Seshadhri. From sylvester-gallai configurations to rank bounds: Improved black-box identity test for depth-3 circuits. *CoRR*, abs/1002.0145, 2010. URL: http://arxiv.org/abs/1002.0145.

23 Robert E. Schapire and Linda M. Sellie. Learning sparse multivariate polynomials over a field with queries and counterexamples. In *Proceedings of the Sixth Annual ACM Workshop on Computational Learning Theory*, pages 17–26, 1996.

24 Amir Shpilka. Interpolation of depth-3 arithmetic circuits with two multiplication gates. In *STOC'07: Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 284–293. ACM Press, 2007.

25 Amir Shpilka and Ilya Volkovich. Improved polynomial identity testing for read-once formulas. In *APPROX-RANDOM*, pages 700–713, 2009.

A Characterizing $\Pi\Sigma$ polynomials (Brill's Equations)

In this section we will explicitly compute a set of polynomials whose common solutions characterize the coefficients of all homogeneous $\Pi\Sigma_{\mathbb{C}}[x_1, \ldots, x_r]$ polynomials of degree d. A clean mathematical construction is given by Brill's Equations given in Chapter 4, [8]. However we still need to calculate the time complexity. But before that we define some operations on polynomials and calculate the time taken by the operation along with the size

of the output. Note that all polynomials are over the field of complex numbers $\mathbb{C}$ and all computations are also done for the complex polynomial rings.

Let $\bar{x} = (x_1, \ldots, x_r)$ and $\bar{y} = (y_1, \ldots, y_r)$ be variables. For any homogeneous polynomial $f(\bar{x})$ of degree d, define

$$f_{\bar{x}^k}(\bar{x}, \bar{y}) = \frac{(d-k)!}{d!}(\sum_i x_i \frac{\partial}{\partial y_i})^k f(\bar{y})$$

Expanding $(\sum_i x_i \frac{\partial}{\partial y_i})^k$ as a polynomial of differentials takes $O((r+k)^r)$ time and has the same order of terms in it. $f(\bar{y})$ has $O((r+k)^r)$ terms. Taking partial derivatives of each term takes constant time and therefore overall computing $(\sum_i x_i \frac{\partial}{\partial y_i})^k f(\bar{y})$ takes $O((r+k)^{2r})$ time. Also the expression obtained will have atmost $O((r+k)^{2r})$ terms. Computing the external factor takes $poly(d)$ time and so for an arbitrary $f(\bar{x})$ computing all $f_{\bar{x}^k}(\bar{x}, \bar{y})$ for $0 \leq k \leq d$ takes $poly((r+d)^r)$ time and has $poly((r+d)^r)$ terms in it. From Section E., Chapter 4 in [8] we also know that $f_{\bar{x}^k}(\bar{x}, \bar{y})$ is a bihomogeneous form of degree k in $\bar{x}$ and degree $d-k$ in $\bar{y}$. It is called the k^{th} polar of f.

Next we define an $\odot$ opeartion between homogeneous forms. Let $f(\bar{x})$ and $g(\bar{x})$ be homogeneous polynomials of degrees d, define

$$(f \odot g)(\bar{x}, \bar{y}) = \frac{1}{d+1} \sum_{k=0}^{d} (-1)^k \binom{d}{k} f_{\bar{y}^k}(\bar{y}, \bar{x}) g_{\bar{x}^k}(\bar{x}, \bar{y})$$

From the discussion above we know that computing $f_{\bar{y}^k}(\bar{y}, \bar{x}) g_{\bar{x}^k}(\bar{x}, \bar{y})$ takes $poly((r+d)^r)$ time and it is obvious that this product has $poly((r+d)^r)$ terms. Rest of the operations take $poly(d)$ time and therefore computing $(f \odot g)(\bar{x}, \bar{y})$ takes $poly((r+d)^r)$ time and has $poly((r+d)^r)$ terms. From the discussion before we may also easily conclude that the degrees of $\bar{x}, \bar{y}$ in $(f \odot g)(\bar{x}, \bar{y})$ are $poly(d)$. The form $(f \odot g)$ is called the vertical(Young) product of f and g. See Section G., Chapter 4 in [8].

Next for $k \in \{0, \ldots, d\}$ and $\bar{z} = (z_1, \ldots, z_r)$ consider homogeneous forms:

$$e_k = \binom{d}{k} f_{\bar{x}^k}(\bar{x}, \bar{z}) f(\bar{z})^{k-1}$$

Following arguments from above, it's straightforward to see that computing e_k takes $poly((r+d)^r)$ time and has $poly((r+d)^r)$ terms. Each e_k is a homogeneous form in $\bar{x}, \bar{z}$ and f. It has degree k in $\bar{x}$, degree $k(d-1)$ in z, and k in coefficients of f. See Section H. of Chapter 4 in [8]. Let's define the following function of $\bar{x}$ with parameters f, z

$$P_{f,z}(\bar{x}) = (-1)^d d \sum_{i_1 + 2i_2 + \ldots + ri_r = d} (-1)^{(i_1 + \ldots + i_r)} \frac{(i_1 + \ldots + i_r - 1)!}{i_1! \ldots i_r!} e_1^{i_1} \ldots e_r^{i_r}$$

Note that $\{(i_1, \ldots, i_r) : i_1 + 2i_2 + \ldots + ri_r = d\} \subseteq \{(i_1, \ldots, i_r) : i_1 + i_2 + \ldots + i_r \leq d\}$ and therefore the number of additions in the above summand is $O(poly(r+d)^r)$. For every fixed $(i_1, \ldots, i_r)$ computing the coefficient $\frac{(i_1 + \ldots + i_r - 1)!}{i_1! \ldots i_r!}$ takes $O(poly((r+d)^r))$ time using multinomial coefficients. Each e_k takes $poly((r+d)^r)$ time to compute. There are r of them in each summand and so overall we take $O(poly((r+d)^r))$ time. A similar argument shows that number of terms in this polynomial is $O(poly((r+d)^r))$. Some more analysis shows

that $P_{f,z}(\bar{x})$ is a form of degree d in $\bar{x}$ whose coefficients are homogeneous polynomials of dedgree d in f and degree $d(d-1)$ in $\bar{z}$. Let

$$B_f(\bar{x},\bar{y},\bar{z}) = (f \odot P_{f,z})(\bar{x},\bar{y})$$

By the arguments given above calculating this form also takes time $poly((r+d)^r)$ and it has $poly((r+d)^r)$ terms. This is a homogeneous form in $(\bar{x},\bar{y},\bar{z})$ of multidegree $(d,d,d(d-1))$ and it's coefficients are forms of degree $(d+1)$ in the coefficients of f. See Section H., Chapter 4 in [8]. So in time $poly((r+d)^r)$ we can compute $B_f(\bar{x},\bar{y},\bar{z})$ explicitly.

Now we arrive at the main theorem

▶ **Theorem A.1** (Brill's Equation – see 4.H, [8]). *A form $f(\bar{x})$ is a product of linear forms if and only if the polynomial $B_f(\bar{x},\bar{y},\bar{z})$ is identically 0.*

We argued above that computing $B_f(\bar{x},\bar{y},\bar{z})$ takes $O(poly((r+d)^r))$ time. It's degrees in $\bar{x},\bar{y},\bar{z}$ are all $poly(d)$ and so the number of coefficients when written as a polynomial over the $3r$ variables

$(x_1,\ldots,x_r,y_1,\ldots,y_r,z,\ldots,z_r)$ is $poly((r+d)^r)$. We mentioned that each coefficient is a polynomial of degree $(d+1)$ in the coefficients of f. Therefore we have the following corollary.

▶ **Corollary A.2.** *Let*

$$I \stackrel{def}{=} \{(\alpha_1,\ldots,\alpha_n) : \forall i : \alpha_i \geq 0, \sum_{i\in[r]} \alpha_i = d\}$$

be the set capturing the indices of all possible monomials of degree exactly d in r variables $(x_1,\ldots,x_r)$. Let $f_{\mathbf{a}}(y_1,\ldots,y_r) = \sum_{\alpha\in I} a_\alpha \mathbf{y}^\alpha$ denote an arbitrary homogeneous polynomial. The coefficient vector then becomes $\mathbf{a} = (a_\alpha)_{\alpha\in I}$. Then there exists an explicit set of polynomials $F_1(\mathbf{a}),\ldots,F_m(\mathbf{a})$ on $poly((r+d)^r)$ variables ($\mathbf{a} = (a_\alpha)_{\alpha\in I}$), with $m = poly((r+d)^r)$, $deg(F_i) \leq poly(d)$ such that for any particular value of $\mathbf{a}$, the corresponding polynomial $f_{\mathbf{a}}(\mathbf{y}) \in \Pi\Sigma^d_{\mathbb{F}}[\bar{y}]$ if and only if $F_1(\mathbf{a}) = \ldots = F_m(\mathbf{a}) = 0$. Also this set $\{F_i, i \in [m]\}$ can be computed in time $poly((r+d)^r)$ time.

Proof. Clear from the theorem and discussion above. ◀

Note that in our application $r = O(1)$ and so $poly((d+r)^r) = poly(d)$.

B Tools from Incidence Geometry

Later in the paper we will use the quantitative version of Sylvester-Gallai Theorem from [3]. In this subsection we do preparation for the same. Our main application will also involve a corollary we prove towards the end of this subsection.

▶ **Definition B.1** ([3]). Let S be a set of n distinct points in complex space $\mathbb{C}^r$. A $k-flat$ is elementary if its intersection with S has exactly $k+1$ points.

▶ **Definition B.2** ([3]). Let S be a set of n distinct points in $\mathbb{C}^r$. S is called a $\delta - SG_k$ configuration if for every independent $s_1,\ldots,s_k \in S$ there are atleast δn points $t \in S$ such that either $t \in fl(\{s_1,\ldots,s_k\})$ or the $k-$flat $fl(\{s_1,\ldots,s_k,t\})$ contains a point in $S \setminus \{s_1,\ldots,s_k,t\}$.

▶ **Theorem B.3** ([3]). *Let S be a $\delta - SG_k$ configuration then $dim(S) \leq \frac{2^{C^k}}{\delta^2}$. Where $C > 1$ is a universal constant.*

This bound on the dimension of S was further improved by Dvir et al. in [6]. The latest version now states

▶ **Theorem B.4** ([6]). *Let S be a $\delta - SG_k$ configuration then $dim(S) \leq C_k = \frac{C^k}{\delta}$. Where $C > 1$ is a universal constant.*

▶ **Corollary B.5.** *Let $dim(S) > C_k$ then S is not a $\delta - SG_k$ configuration i.e. there exists a set of independent points $\{s_1, \ldots, s_k\}$ and $\geq (1-\delta)n$ points t such that $fl(\{s_1, \ldots, s_k, t\})$ is an elementary $k - flat$. That is:*

- $t \notin fl(\{s_1, \ldots, s_k\})$
- $fl(\{s_1, \ldots, s_k, t\}) \cap S = \{s_1, \ldots, s_k, t\}$.

Right now we set δ to be a constant $< 0.5, C_k = \frac{C^k}{\delta}$. Note that $C_i < C_{i+1}$. Using the above theorem we prove the following lemma which will be useful to us later

▶ **Lemma B.6** (Bichromatic semi-ordinary line). *Let X and Y be disjoint finite sets in $\mathbb{C}^r$ satisfying the following conditions.*

1. *$dim(Y) > C_4$.*
2. *$|Y| \leq c|X|$ with $c < \frac{1-\delta}{\delta}$.*

Then there exists a line l such that $|l \cap Y| = 1$ and $|l \cap X| \geq 1$

Proof. We consider two cases:

Case 1: $c|X| \geq |Y| \geq |X|$ Since $dim(Y) > C_1$, using the corollary above for $S = X \cup Y, k = 1$ we can get a point $s_1 \in X \cup Y$ for which there exist $(1-\delta)(|X| + |Y|)$ points t in $X \cup Y$ such that $t \notin fl\{s_1\}$ and $fl\{s_1, t\}$ is elementary. If $s_1 \in X$ then $(1-\delta)(|X| + |Y|) - |X| \geq (1-2\delta)|X| > 0$ of these flats intersect Y and thus we get such a line l. If $s_1 \in Y$ then $(1-\delta)(|X| + |Y|) - |Y| \geq ((1-\delta)(\frac{1}{c} + 1) - 1)|Y| > 0$ of these flats intersect X giving us the required line l with $|l \cap X| = 1$ and $|l \cap Y| = 1$.

Case 2: $|Y| \leq |X|$ Now choose a subset $X_1 \subseteq X$ such that $|X_1| = |Y|$. Now using the same argument as above for $S = X_1 \cup Y$ there is a point $s_1 \in X_1 \cup Y$ such that $(1-\delta)(|X_1| + |Y|) = 2(1-\delta)|Y| = 2(1-\delta)|X_1|$ flats through it are elementary in $X_1 \cup Y$. If $s_1 \in Y$ $(1-2\delta)|Y| > 0$ of these flats intersect X_1. If $s_1 \in X_1$, $(1-2\delta)|X_1| > 0$ of these flats intersect Y. In both these above possibilities the flat intersects Y and X_1 in exactly one point each. But it may contain more points from $X \setminus X_1$ so we can find a line l such that $|l \cap Y| = 1$ and $|l \cap X| \geq 1$. ◀

C A Method of Reconstructing Linear Forms

In a lot of circumstances one might reconstruct a linear form (upto scalar multiplication) inside $V = Lin_{\mathbb{F}}[\bar{x}]$ from it's projections (upto scalar multiplication) onto some subspaces of V. For example consider a linear form $L = a_1x_1 + a_2x_2 + a_3x_3 (\in Lin_{\mathbb{F}}[x_1, x_2, x_3])$ with $a_3 \neq 0$, and assume we know scalar multiples of projections of L onto the spaces $\mathbb{F}x_1$ and $\mathbb{F}x_2$ i.e. we know $L_1 = \alpha(a_2x_2 + a_3x_3)$ and $L_2 = \beta(a_1x_1 + a_3x_3)$ for some $\alpha, \beta \in \mathbb{F}$. Scale these projections to $\tilde{L}_1 = x_3 + \frac{a_2}{a_3}x_3$ and $\tilde{L}_2 = x_3 + \frac{a_1}{a_3}x_3$. Using these two define a linear form $x_3 + \frac{a_1}{a_3}x_1 + \frac{a_2}{a_3}x_2$. This is a scalar multiple of our original linear form L. We generalize this a little more below.

Let $\bar{x} \equiv (x_1, \ldots, x_r)$, $\mathcal{B} = \{l_1, \ldots, l_r\}$ be a basis for $V = Lin_{\mathbb{F}}[x_1, \ldots, x_r]$. For $i \in \{0, 1, 2\}$, let S_i be pairwise disjoint non empty subsets of $\mathcal{B}$ such that $S_0 \cup S_1 \cup S_2 = \mathcal{B}$. Let $W_i = sp(S_i)$ and $W_i' = \bigoplus_{j \neq i} W_j$. Clearly $V = W_0 \oplus W_1 \oplus W_2 = W_i \oplus W_i', i \in \{0, 1, 2\}$.

▶ **Lemma C.1.** *Assume $L \in V$ is a linear form such that*
- $\pi_{W_2}(L) \neq 0$
- *For $i \in \{0,1\}, L_i = \beta_i \pi_{W_i'}(L)$ are known for some non-zero scalars β_i.*

Then L is unique upto scalar multiplication and we can construct a scalar multiple $\tilde{L}$ of L.

Proof. Let $L = a_1 l_1 + \ldots + a_r l_r, a_i \in \mathbb{F}$. Since $\pi_{W_2}(L) \neq 0$, there exists $l_j \in S_2$ such that $a_j \neq 0$. Let $\tilde{L} = \frac{1}{a_j} L$. For $i \in \{0,1\}$, re-scale L_i to get $\tilde{L_i}$ making sure that coefficient of l_j is 1 in them. Thus for $i = 0, 1$

$$\pi_{W_i'}(\tilde{L}) = \tilde{L_i}\,.$$

Since $W_0' = W_1 \oplus W_2$ and $W_1' = W_0 \oplus W_2$ by comparing coefficients we can get $\tilde{L}$. ◀

Algorithm Assume we know S_0, S_1, S_2 and therefore the basis change matrix to convert vector representations from $\mathcal{S}$ to $\mathcal{B}$. It takes $poly(r)$ time to convert $[v]_{\mathcal{S}}$ to $[v]_{\mathcal{B}}$. Given L_i in the basis $\mathcal{B}$ it takes $poly(r)$ time(by a linear scan) to find $l_j \in S_2$ with $a_j \neq 0$. This l_j has a non zero coefficient in both L_0, L_1. After this we just rescale L_i to get $\tilde{L_i}$ such that coefficient of l_j is 1. Then since $\tilde{L_i} = \pi_{W_i'}(\tilde{L})$ the coefficient of l_t in $\tilde{L}$ is as follows:

$$= \begin{cases} \text{coefficient of } l_t \text{ in } \tilde{L_1} & : l_t \in S_0 \\ \text{coefficient of } l_t \text{ in } \tilde{L_0} & : l_t \in S_1 \\ \text{coefficient of } l_t \text{ in } \tilde{L_0} = \text{ coefficient of } l_t \text{ in } \tilde{L_1} & : l_t \in S_2 \end{cases}$$

Finding the right coefficients using this also takes $poly(r)$ time.

Next we try and use this to reconstruct $\Pi\Sigma$ polynomials. This case is slightly more complicated and so we demand that the projections have some special form. In particular the projections onto one subspace preserves pairwise linear independence of linear factors and onto the other makes all linear factors scalar multiples of each other.

▶ **Corollary C.2.** *Let $S_i, W_i, i \in \{0,1,2\}$ be as above and $P \in \Pi\Sigma_{\mathbb{F}}[x_1, \ldots, x_r]$ such that*
1. $\pi_{W_2}(P) \neq 0$
2. *For $i \in \{0,1\}$ there exists $\beta_i (\neq 0) \in \mathbb{F}$ such that $P_0 = \beta_0 \pi_{W_0'}(P) = p^t$ and $P_1 = \beta_1 \pi_{W_1'}(P) = d_1 \ldots d_t$. are known i.e. p, d_j $(j \in [t])$ and t are known.*

Then P is unique upto scalar multiplication and we can construct a scalar multiple $\tilde{P}$ of P.

Proof. Let $P = L_1 \ldots L_t$ with $L_i \in Lin_{\mathbb{F}}[\bar{x}]$. There exists $\beta_i^j, i \in \{0,1\}, j \in [t]$, such that $\beta_0^j \pi_{W_0'}(L_j) = p$ and $\beta_1^j \pi_{W_1'}(L_j) = d_j$. Since p, d_j are known by above Lemma C.1 we find a scalar multiple $\tilde{L_j} = \beta^j L_j$ of L_j and therefore find a scalar multiple $\tilde{P} = \tilde{L_1} \ldots \tilde{L_t}$ of P. Note that this method also tells us that such a P is unique upto scalar multiplication. Since we've used the above Algorithm C at most t times with $t \leq deg(P)$, it takes $poly(deg(P), r)$ time to find $\tilde{P}$. ◀

This corollary is the backbone for reconstructing $\Pi\Sigma$ polynomials from their projections. But first we formally define a "*Reconstructor*":

▶ **Definition C.3** (Reconstructor)**.** Let $S_i, W_i, i \in \{0,1,2\}$ be as above. Let Q be a standard $\Pi\Sigma$ polynomial and P be a standard $\Pi\Sigma$ polynomial dividing Q with $Q = PR$. Then (Q, P, S_0, S_1, S_2) is called a *Reconstructor* if:
- $\pi_{W_2}(P) \neq 0$.
- $\pi_{W_0'}(P) = \alpha p^t$, for some linear form p.
- Let $l \mid R$ be a linear form and $\pi_{W_2}(l) \neq 0$ then $gcd(\pi_{W_2}(P), \pi_{W_2}(l)) = 1$.

Note: Let L_1, L_2 be two LI linear forms dividing P , then one can show

L_1, L_2 are LI $\Leftrightarrow \pi_{W_1'}(L_1), \pi_{W_1'}(L_2)$ are LI

To see this first observe that the second bullet implies for $i \in [2], L_i \in W_0 + p \Rightarrow sp(\{L_1, L_2\}) \subseteq W_0 + p$.

If $\pi_{W_1'}(L_1), \pi_{W_1'}(L_2)$ are LD then

$sp(\{L_1, L_2\}) \cap W_1 \neq \{0\}$

$\Rightarrow (W_0 + p) \cap W_1 \neq \{0\}$. Since $W_0 \cap W_1 = \{0\}$ we get that $p \in W_0 \oplus W_1 = W_2' \Rightarrow \pi_{W_2}(p) = 0 \Rightarrow \pi_{W_2}(P) = 0$ contradicting the first bullet.

Geometrically the conditions just mean that all linear forms dividing P have LD projection ($= \gamma p$ for some non zero $\gamma \in \mathbb{F}$) w.r.t. the subspace W_0' and LI linear forms p_1, p_2 dividing P have LI projections (w.r.t. subspace W_1'). Also no linear form l dividing R belongs to $fl(S_0 \cup S_1 \cup \{p\})$.

We are now ready to give an algorithm to reconstruct P using $\pi_{W_0'}(Q)$ and $\pi_{W_1'}(Q)$ by gluing appropriate projections corresponding to P. To be precise:

▶ **Claim C.4.** *Let Q, P be standard $\Pi\Sigma$ polynomials and $P \mid Q$. Assume (Q, P, S_0, S_1, S_2) is a* Reconstructor. *If we know both $\pi_{W_0'}(Q)$ and $\pi_{W_1'}(Q)$. Then we can reconstruct P.*

Proof. See Algorithm 7. ◀

C.1 Explanation

- The algorithm takes as input projections $\pi_{W_0'}(Q)$ and $\pi_{W_1'}(Q)$ along with the sets $S_i, i = 0, 1, 2$ which form a partition of a basis $\mathcal{B}$. We know that there exists a polynomial $P \mid Q$ such that (Q, P, S_0, S_1, S_2) is a reconstructor and so we try to compute the projections $\pi_{W_0'}(P), \pi_{W_1'}(P)$.
- If one assumes that $\pi_{W_0'}(Q) = \gamma \prod_{i \in [s]} c_i^{m_i}$ with the c_i's co-prime, then by the properties of a reconstructor the projection (of a scalar multiple of P) onto W_0' say $P_0 = \beta_0 \pi_{W_0'}(P)$ (for some β_0) has to be equal to $c_i^{m_i}$ for some i. We do this assignment inside the first for loop.
- The third property of a reconstructor implies that when we project further to W_2, it should not get any more factors and so we check this inside the second for loop by going over all other factors c_j of $\pi_{W_0'}(Q)$ and checking if c_i, c_j become LD on projecting to W_2 (i.e. by further projecting to W_1').
- Now to find (scalar multiple of) the other projections i.e. $P_1 = \beta_1 \pi_{W_1'}(P)$ (for some β_1), we go through $\pi_{W_1'}(Q)$ and find d_k such that $\{\pi_{W_1'}(c_i), \pi_{W_0'}(d_k)\}$ are LD (i.e. they are projections of the same linear form). We collect the product of all such d_k's. If the choice of c_i were correct then all d_k's would be obtained correctly.
- The last "*if*" statement just checks that the number of d_k's found above is the same as m_i since $P_0 = c_i^{m_i}$ tells us that the degree of P was m_i. We recover a scalar multiple of P using the algorithm explained in Corollary C.2 and then make it standard to get P.

C.2 Correctness

The corectness of our algorithm is shown by the lemma below.

input : $\pi_{W_0'}(Q) \in \Pi\Sigma[\bar{x}], \pi_{W_1'}(Q) \in \Pi\Sigma[\bar{x}], S_0, S_1, S_2$
output : a $\Pi\Sigma$ polynomial $P \mid Q$

$bool flag$, $\Pi\Sigma$ polynomial P_0, P_1;;
Factor $\pi_{W_0'}(Q) = \gamma \prod_{i\in[s]} c_i^{m_i}$, c_i's pairwise LI and normal, $\gamma \in \mathbb{F}$;
Factor $\pi_{W_1'}(Q) = \delta d_1 \dots d_m$, $\delta \in \mathbb{F}$ and d_j normal;
for $i \in [s]$ *&&* $\pi_{W_1'}(c_i) \neq 0$ **do**
  $flag = true, P_0 = c_i^{m_i}$;
  `// Assuming projection w.r.t.` W_0' `to be` $c_i^{m_i}$.
  for $j \in [s]$ *&&* $j \neq i$ *&&* $\pi_{W_1'}(c_j) \neq 0$ **do**
    if $gcd(\pi_{W_1'}(c_i), \pi_{W_1'}(c_j)) \neq 1$ **then**
      $flag = false$;
    end
  end
  if $flag == true$ **then**
    $P_1 = 1$;
  end
  for $j \in [m]$ **do**
    if $\pi_{W_0'}(d_j) \neq 0$ *& &* $\{\pi_{W_0'}(d_j), \pi_{W_1'}(c_i)\}$ *are LD* **then**
      $P_1 = P_1 d_j$;
      `// This steps collects projection w.r.t.` W_1' `in` P_1.
    end
  end
  if $(deg(P_1) = m_i)$ *&&* (P_0, P_1) *give* $\tilde{P} = \beta P$ *using Corollary C.2* **then**
    Make $\tilde{P}$ *standard* w.r.t. the standard basis $\mathcal{S}$ to get P;
    Return P;
  end
end
Return 1;

Algorithm 7: Reconstructing Linear Factors.

▶ **Claim C.5.** *If (Q, P, S_0, S_1, S_2) is a reconstructor for non-constant P, then Algorithm refreconalgo returns P.*

Proof. (Q, P, S_0, S_1, S_2) is a reconstructor therefore

- $\pi_{W_2}(P) \neq 0$
- $\pi_{W_0'}(P) = \delta p^t$
- $q \mid \frac{Q}{P} \Rightarrow gcd(\pi_{W_2}(q), \pi_{W_2}(P)) = 1$

1. It is clear that for one and only one value of i, c_i divides p. Fix this i. Let $Q = PR$, if $c_i^{m_i} \nmid \pi_{W_0'}(P)$ then $c_i \mid l$ for some linear form $l \mid \pi_{W_0'}(R)$. Condition 3 in definition of Reconstructor implies that $gcd(\pi_{W_2}(P), \pi_{W_2}(l)) = 1$ but $\pi_{W_2}(c_i)$ divides both of them giving us a contradiction. Since $\pi_{W_0'}(P)$ has just one linear factor $\Rightarrow \pi_{W_0'}(P)$ is a scalar multiple of $c_i^{m_i}$ for some i.
2. Assume the correct $c_i^{m_i}$ has been found. Now let $d_j \mid \pi_{W_1'}(Q)$ such that $\{\pi_{W_2}(c_i), \pi_{W_2}(d_j)\}$ are LD. then we can show that $d_j \mid \pi_{W_1'}(P)$. Assume not, then for some linear form $l \mid R = \frac{Q}{P}$, $d_j \mid \pi_{W_1'}(l)$. $\pi_{W_0'}(d_j) \neq 0$ (which we checked) $\Rightarrow \pi_{W_2}(l) \neq 0$. So we get

$\pi_{W_2}(c_i) \mid \pi_{W_2}(l)(\neq 0)$ and so $\pi_{W_2}(c_i) \mid gcd(\pi_{W_2}(P), \pi_{W_2}(l))$ which is therefore $\neq 1$ and condition 3 of Definiton C.3 is violated. So whatever d_j we collect will be a factor of $\pi_{W_1'}(P)$ and we will collect all of them since they are all present in $\pi_{W_1'}(Q)$.

3. We know from proof of Corollary C.2 that if we know c_i, m_i and d_j's correctly then we can recover a scalar multiple of P correctly. But Q, P are standard so we return P correctly. ◀

In fact we can show that if we return something it has to be a factor of Q.

▶ **Claim C.6.** *If Algorithm 7 returns a* $\Pi\Sigma$ *polynomial* P*, then* $P \mid Q$*.*

- If the algorithm returns 1 from the last return statement, we are done. So let's assume it returns something from the previous return statement.
- So $flag$ has to be true at end $\Rightarrow$ there is an $i \in [s]$ such that $P_0 = c_i^{m_i}$ with the conditions that $\pi_{W_1'}(c_i) \neq 0$ and $gcd(c_i, c_j) = 1$ for $j \neq i$. It also means that for exactly m_i of the d_j's (say $d_1, \ldots, d_{m_i}$) $\{\pi_{W_1'}(c_i), \pi_{W_0'}(d_j)\}$ are LD and $P_1 = d_1 \ldots d_{m_i}$.
- Since $c_i^{m_i} \mid \pi_{W_0'}(Q)$, there exists a factor $\tilde{P} \mid Q$ of degree m_i such that $\pi_{W_0'}(\tilde{P}) = c_i^{m_i}$ and $\pi_{W_1'}(c_i) \neq 0$. This $\Rightarrow \pi_{W_2}(\tilde{P}) \neq 0$. Clearly $\pi_{W_1'}(\tilde{P}) \mid \pi_{W_1'}(Q) = d_1 \ldots d_m$, hence for all linear factors $\tilde{p}$ of $\tilde{P}$, $\pi_{W_1'}(\tilde{p})$ should be some d_j with the condition that $\{\pi_{W_0'}((\pi'_{W_1})(\tilde{p})), \pi_{W_1'}(c_i)\}$ should be LD. The only choice we have are $d_1, \ldots, d_{m_i}$. So $\pi_{W_0'}(\tilde{P}) = d_1 \ldots d_{m_i}$. All conditions of Corollary C.2 are true and so $\tilde{P}$ is uniquely defined (upto scalar multiplication) by the reconstruction method given in Corollary C.2. So what we returned was actually a factor of Q.

C.3 Time Complexity

Factoring $\pi_{W_0'}(Q), \pi_{W_1'}(Q)$ takes $poly(d)$ time (using Kaltofen's Factoring from [14]). The nested for loops run $\leq d^3$ times. Computing projections with respect to the known decomposition $W_0 \oplus W_1 \oplus W_2 = \mathbb{F}^r$ of linear forms over r variables takes $poly(r)$ time. Computing gcd and linear independence of linear forms takes $poly(r)$ time. The final reconstruction of P using (P_0, P_1) takes $poly(d, r)$ time as has been explained in Corollary C.2. Multiplying linear forms to $\Pi\Sigma$ polynomial takes $poly(d^r)$ time. Therefore overall the algorithm takes $poly(d^r)$ time. In our application $r = O(1)$ and therefore the algorithm takes $poly(d)$ time.

D Random Linear Transformations

This section will prove some results about linear independence and non-degeneracy under random transformations on $\mathbb{F}^r$. This will be required to make our input non-degenerate. From here onwards we fix a natural number $N \in \mathbb{N}$ and assume $0 < k < r$. Let $T \subset \mathbb{F}^r$ be a finite set with $dim(T) = r$. Next we consider two $r \times r$ matrices Ω, Λ. Entries $\Omega_{i,j}, \Lambda_{i,j}$ are picked independently from the uniform distribution on $[N]$. For any basis $\mathcal{B}$ of $\mathbb{F}^r$ and vector $v \in \mathbb{F}^r$, let $[v]_{\mathcal{B}}$ denote the co-ordinate vector of v in the basis $\mathcal{B}$. If $\mathcal{B} = \{b_1, \ldots, b_r\}$ then $[v]^i_{\mathcal{B}}$ denotes the i-th co-ordinate in $[v]_{\mathcal{B}}$. Let $\mathcal{S} = \{e_1, \ldots, e_r\}$ be the standard basis of $\mathbb{F}^r$. Let $E_j = sp(\{e_1, \ldots, e_j\})$ and $E'_j = sp(\{e_{j+1}, \ldots, e_r\})$, then $\mathbb{F}^r = E_j \oplus E'_j$. Let $\pi_{W_{E_j}}$ be the orthogonal projection onto E_j. For any matrix M, we denote the matrix of it's co-factors by $co(M)$. We consider the following events:

- $\mathcal{E}_0 = \{\Omega$ is not invertible $\}$
- $\mathcal{E}_1 = \{\exists t(\neq 0) \in T : \pi_{W_{E_1}}(\Omega(t)) = 0\}$
- $\mathcal{E}_2 = \{\exists \{t_1, \ldots, t_r\}$ LI vectors in $T : \{\Omega(t_1), \ldots, \Omega(t_r)\}$ is LD $\}$
- $\mathcal{E}_3 = \{\exists \{t_1, \ldots, t_r\}$ LI vectors in $T : \{\Omega(t_1), \ldots, \Omega(t_k), \Lambda\Omega(t_{k+1}), \ldots, \Lambda\Omega(t_r)\}$ is LD $\}$

- When t_i, Λ, Ω are clear we define the matrix $M = [M_1 \dots M_r]$ with columns M_i given as:

$$M_i = \begin{cases} [\Omega(t_i)]_{\mathcal{S}} : i \leq k \\ [\Lambda\Omega(t_i)]_{\mathcal{S}} : i > k \end{cases}$$

  M corresponds to the linear map

  $e_i \mapsto \Omega(t_i)$ for $i \leq k$ and $e_i \mapsto \Lambda\Omega(t_i)$ for $i > k$

- $\mathcal{E}_4 = \{\{\exists\{t_1, \dots, t_r\}$ LI vectors in T and $t \in T \setminus sp(\{t_1, \dots, t_k\}) : [co(M)[\Omega(t)]_{\mathcal{S}}]_{\mathcal{S}}^{k+1} = 0\}$
- $\mathcal{E}_5 = \mathcal{E}_4 \mid \mathcal{E}_3^c$

Next we show that the probability of all of the above events is small. Before doing that let's explain the events. This will give an intuition to why the events have low probabilities.

- $\mathcal{E}_0$ is the event where Ω is not-invertible. Random Transformations should be invertible.
- $\mathcal{E}_1$ is the event where there is a non-zero $t \in T$ such that the projection to the first co-ordinate (w.r.t. $\mathcal{S}$) of Ω applied on t is 0. We don't expect this for a random linear transformation. Random Transformation on a non-zero vector should give a non-zero coefficient of e_1.
- $\mathcal{E}_2$ is the event such that Ω takes a basis to a LD set i.e. Ω is not invertible (random linear operators are invertible).
- $\mathcal{E}_3$ is the event such that for some basis applying Ω to the first k vectors and $\Lambda\Omega$ to the last $n-k$ vectors gives a LD set. So this operation is not-invertible. For ranrom maps this should not be the case.
- $\mathcal{E}_4$ is the event that there is some basis $\{t_1, \dots, t_r\}$ and t outside $sp(t_1, \dots, t_k)$ such that the $(k+1)^{th}$ co-ordinate of $co(M)[\Omega(t)]_{\mathcal{S}}$ w.r.t the standard basis is 0. If M were invertible, clearly the set $\mathcal{B} = \{\Omega(t_1), \dots, \Omega(t_k), \Lambda\Omega(t_{k+1}), \dots, \Lambda\Omega(t_r)\}$ would be a basis and $co(M)$ will be a scalar multiple of M^{-1}. So we are asking if the $(k+1)^{th}$ co-ordinate of $\Omega(t)$ in the basis $\mathcal{B}$ is 0. For random Ω, Λ we would expect M to be invertible and this co-ordinate to be non-zero.

Now let's formally prove everything. We will repeatedly use the popular Schawrtz-Zippel Lemma which the reader can find in [21].

▶ **Claim D.1.** $Pr[\mathcal{E}_1] \leq \frac{|T|}{N}$.

Proof. Fix a non-zero $t = \begin{pmatrix} a_1 \\ . \\ . \\ a_r \end{pmatrix}$ with $a_i \in \mathbb{F}$ and let $\Omega = (\Omega_{i,j}), 1 \leq i, j \leq r$. Then the first co-ordinate of $\Omega(t)$ is $\Omega_{1,1}a_1 + \Omega_{1,2}a_2 + \dots + \Omega_{1,r}a_r$. Since $t \neq 0$, not all a_i are 0 and this is therefore not an identically zero polynomial in $(\Omega_{1,1}, \dots, \Omega_{1,r})$. Therefore by Schwartz-Zippel lemma $Pr[[\Omega(t)]_{\mathcal{S}}^1 = 0] \leq \frac{1}{N}$. Using a union bound inside T we get $Pr[\exists t(\neq 0) \in T : [\Omega(t)]_{\mathcal{S}}^1 = 0] \leq \frac{|T|}{N}$. ◀

▶ **Claim D.2.** $Pr[\mathcal{E}_2] \leq \frac{r}{N}$.

Proof. Clearly $\mathcal{E}_2 \subseteq \mathcal{E}_0$ and so $Pr[\mathcal{E}_2] \leq Pr[\mathcal{E}_0]$. $\mathcal{E}_0$ corresponds to the polynomial equation $det(\Omega) = 0$. $det(\Omega)$ is a degree r polynomial in r^2 variables and is also not identically zero, so using Schwartz-Zippel lemma we get $Pr[\mathcal{E}_2] \leq Pr[\mathcal{E}_0] \leq \frac{r}{N}$. ◀

▶ **Claim D.3.** $Pr[\mathcal{E}_3] \leq \binom{|T|}{r}\frac{2r}{N}$.

Proof. Fix an LI set $t_1, \ldots, t_r$. The set $\{\Omega(t_1), \ldots, \Omega(t_k), \Lambda\Omega(t_{k+1}), \ldots \Lambda\Omega(t_r)\}$ is LD iff the $r \times r$ matrix M formed by writing these vectors (in basis $\mathcal{S}$) as columns (described in part D above) has determinant 0. M has entries polynomial (of degree ≤ 2) in $\Omega_{i,j}$ and $\Lambda_{i,j}$ and so $det(M)$ is a polynomial in $\Omega_{i,j}, \Lambda_{i,j}$ of degree $\leq 2r$. For $\Omega = \Lambda = I$ (identity matrix) this matrix just becomes the matrix formed by the basis $\{t_1, \ldots, t_r\}$ which has non-zero determinant and so $det(M)$ is not the identically zero polynomial. By Schwartz-Zippel lemma $Pr[det(M) = 0] \leq = \frac{2r}{N}$. Now we vary the LI set $\{t_1, \ldots, t_r\}$, there are $\leq \binom{|T|}{r}$ such sets and so by a union bound $Pr[\mathcal{E}_3] \leq \binom{|T|}{r}\frac{2r}{N}$. ◀

▶ **Claim D.4.** $Pr[\mathcal{E}_4] \leq \binom{|T|}{r+1}\frac{2r-1}{N}$.

Proof. Fix an LI set $t_1, \ldots, t_r$ and a vector $t \notin sp(\{t_1, \ldots, t_k\})$. Let $t = \sum_{i=1}^{r} a_i t_i$. Since $t \notin sp(\{t_1 \ldots, t_k\})$, $a_s \neq 0$ for some $s \in \{k+1, \ldots, r\}$. Let $\mathcal{B} = \{\Omega(t_1), \ldots, \Omega(t_k), \Lambda\Omega(t_{k+1}), \ldots \Lambda\Omega(t_r)\}$. Let M be the matrix whose columns are from $\mathcal{B}$ (Construction has been explained in part D above). We know that the co-factors of a matrix are polynomials of degree $\leq r-1$ in the matrix elements. In our matrix M all entries are polynomials of degree ≤ 2 in $\Omega_{i,j}, \Lambda_{i,j}$, so all entries of $co(M)$ are polynomials of degree $\leq 2r-2$ in $\Omega_{i,j}, \Lambda_{i,j}$. Thus $[co(M)[\Omega(t)]_{\mathcal{S}}]_{\mathcal{S}}^{k+1} = \sum_{i=1}^{r} co(M)_{k+1,i}[\Omega(t)]_{\mathcal{S}}^{i}$ is a polynomial of degree $\leq 2r-1$. This polynomial is not identically zero. Define Ω to be the matrix (w.r.t. basis $\mathcal{S}$) of the linear map $\Omega(t_i) = e_i$ and Λ to be the matrix (w.r.t. basis $\mathcal{S}$) of the map

$$\Lambda = \begin{cases} \Lambda(e_i) = e_i : i \notin \{s, k+1\} \\ \Lambda(e_s) = e_{k+1} \\ \Lambda(e_{k+1}) = e_s \end{cases}$$

With these values the set $\mathcal{B}$ becomes $\{e_1, \ldots, e_k, e_s, e_{k+2}, \ldots, e_{s-1}, e_{k+1}, e_{s+1}, \ldots, e_r\}$. If one now looks at M i.e. the matrix formed using entries of $\mathcal{B}$ as columns it's just the permutation matrix that flips e_s and e_{k+1}. This matrix is the inverse of itself and so has determinant $= \pm 1$,

thus $co(M) = \pm M^{-1} = \pm M$. Therefore $co(M)[\Omega(t)]_{\mathcal{S}} = \pm M \begin{pmatrix} a_1 \\ . \\ . \\ a_r \end{pmatrix} = \pm \begin{pmatrix} a_1 \\ . \\ a_k \\ a_s \\ a_{k+2} \\ . \\ a_{s-1} \\ a_{k+1} \\ .a_{s+1} \\ . \\ a_r \end{pmatrix}$.

Since $a_s \neq 0$, we get $[co(M)[\Omega(t)]_{\mathcal{S}}]_{\mathcal{S}}^{k+1} \neq 0$. So the polynomial is not identically zero and we can use Schwartz-Zippel Lemma to say that $Pr[[co(M)[\Omega(t)]_{\mathcal{S}}]_{\mathcal{S}}^{k+1} = 0] \leq \frac{2r-1}{N}$. Now we vary $\{t_1, \ldots, t_r, t\}$ inside T and use union bound to show $Pr[\mathcal{E}_4] \leq \binom{|T|}{r+1}\frac{2r-1}{N}$. ◀

Even though this is just basic probability we include the following:

▶ **Claim D.5.** $Pr[\mathcal{E}_5] \leq \binom{|T|}{r}\frac{2r-1}{N-\binom{|T|}{r}2r}$.

Proof. $Pr[\mathcal{E}_5] = Pr[\mathcal{E}_4 \mid \mathcal{E}_3^c] = \frac{Pr[\mathcal{E}_4 \cap \mathcal{E}_3^c]}{Pr[\mathcal{E}_3^c]} \leq \frac{Pr[\mathcal{E}_4]}{Pr[\mathcal{E}_3^c]} \leq \binom{|T|}{r+1}\frac{\frac{2r-1}{N}}{1-\binom{|T|}{r}\frac{2r}{N}} = \binom{|T|}{r+1}\frac{2r-1}{N-\binom{|T|}{r}2r}$. ◀

In our application of the above $r = O(1), |T| = poly(d), N = 2^d$ and so all probabilities are very small as d grows. So we will assume that none of the above events occur. By union bound that too will have small probability and so with very high probability $\mathcal{E}_0, \mathcal{E}_1, \mathcal{E}_2, \mathcal{E}_3, \mathcal{E}_4, \mathcal{E}_5$ do not occur.

E Set $\mathcal{C}$ of Candidate Linear Forms

This section deals with constructing a $poly(d)$ size set $\mathcal{C}$ which contains each $l_{ij}, (i,j) \in \{0,1\} \times [M]$. First we define the set and prove a bound on it's size.

E.1 Structure and Size of $\mathcal{C}$

Let's recall $f = G(\alpha_0 T_0 + \alpha_1 T_1)$ and define two other polynomials:

$$g = \frac{f}{G} = \alpha_0 T_0 + \alpha_1 T_1$$

$$h = \frac{f}{Lin(f)} = \frac{g}{Lin(g)}$$

Assume $deg(h) = d_h$:

▶ **Definition E.1.** Our candidate set is defined as:

$$\mathcal{C} \stackrel{def}{=} \{l = x_1 - a_2x_2 - \ldots - a_rx_r \in Lin_{\mathbb{F}}[\bar{x}] : h(a_2x_2 + \ldots + a_rx_r, x_2, \ldots, x_r) \in \Pi\Sigma^{d_h}_{\mathbb{F}}[x_2, \ldots, x_r]\}$$

(for definition of $\Pi\Sigma^{d_h}_{\mathbb{F}}[x_2, \ldots, x_r]$ see Section 1.4).

In the claim below we show that linear forms dividing polynomials $T_i, i = 0, 1$ are actually inside $\mathcal{C}$ (first part of claim). The remaining linear forms in $\mathcal{C}$ (which we call *"spurious"*) have a nice structure (second part of claim). In the third part of our claim we arrive at a bound on the size of $\mathcal{C}$. Recall the definition of $c_{\mathbb{F}}(k)$ from Theorem 1.7.

▶ **Claim E.2.** *The following are true about our candidate set $\mathcal{C}$.*

1. $\mathcal{L}(T_i) \subseteq \mathcal{C}, i = 0, 1$.
2. *Let $k = c_{\mathbb{F}}(3) + 2$ and suppose $\{l_j; j \in [k]\} \subset \mathcal{L}(T_i)$ are LI. Then for any $l \in \mathcal{C} \setminus (\mathcal{L}(T_0) \cup \mathcal{L}(T_1))$, there exists $j \in [k]$ such that $fl(\{l, l_j\}) \cap \mathcal{L}(T_{1-i}) \neq \phi$ i.e. the line joining l and l_j does not intersect the set $\mathcal{L}(T_{1-i})$.*
3. $|\mathcal{C}| \leq M^4 + 2M \leq d^4 + 2d$.

Proof. Let's first recall the definition of our candidate set

$$\mathcal{C} \stackrel{def}{=} \{l = x_1 - a_2x_2 - \ldots - a_rx_r \in Lin_{\mathbb{F}}[\bar{x}] : h(a_2x_2 + \ldots + a_rx_r, x_2, \ldots, x_r) \in \Pi\Sigma^{d_h}_{\mathbb{F}}[x_2, \ldots, x_r]\}$$

Also recall that

$$h = \frac{g}{Lin(g)} = \frac{f}{Lin(f)}.$$

1. Let $l = x_1 - a_2x_2 - \ldots - a_rx_r \in \mathcal{L}(T_{1-i})$. Let's denote the tuple $v \equiv (a_2x_2 + \ldots + a_rx_r, x_2, \ldots, x_r)$. Since $gcd(T_0, T_1) = 1$ and $l \mid T_{1-i}$ we know that $l \nmid T_i$ and therefore $Lin(g)(v) \neq 0$. We can then compute

   $$h(v) = \frac{\alpha_i T_i(v)}{Lin(g)(v)} = \alpha_i H_1(v) \ldots H_{d_h}(v) \in \Pi\Sigma^{d_h}_{\mathbb{F}}[x_2, \ldots, x_r]$$

   where $H_j \in Lin_{\mathbb{F}}[x_2, \ldots, x_r]$. So $\mathcal{L}(T_i) \subseteq \mathcal{C}$ for $i = 0, 1$.

2. Consider $l = x_1 - a_2x_2 - \ldots - a_rx_r \in \mathcal{C} \setminus (\mathcal{L}(T_0) \cup \mathcal{L}(T_1))$ and assume that $sp(\{l, l_j\}) \cap \mathcal{L}(T_{1-i}) = \phi$ for all $j \in [k]$. We know that

   $$g(v) = Lin(g)(v)H_1(v)\ldots H_{d_h}(v) = \alpha_0 T_0(v) + \alpha_1 T_1(v)\,.$$

   Let g' be the following identically zero $\Sigma\Pi\Sigma(3)[x_2, \ldots, x_r]$ polynomial (with circuit $\mathcal{C}'$)

   $$g' = Lin(g)(v)H_1(v)\ldots H_{d_h}(v) - \alpha_0 T_0(v) - \alpha_1 T_1(v)\,.$$

   We know

   $$\mathcal{C}' = gcd(\mathcal{C}')Sim(\mathcal{C}') \Rightarrow Sim(\mathcal{C}') \equiv 0\,.$$

   Recall that $l_j(v) \mid T_i(v)$, therefore the $l_j(v)$ cannot be factors of $gcd(\mathcal{C}')$ because if they did then there exist pair $l_j, l_{(1-i)t}$ such that $\{l_j(v), l_{(1-i)t}(v)\}$ is LD or in other words $sp(\{l, l_j\}) \cap \mathcal{L}(T_{1-i}) \neq \phi$ and we have a contradiction. Also the set $\{l_j(v) : j \in [k]\}$ has dimension $\geq k-1$ since the dimension could fall only by 1 when we go modulo a linear form (project to hyperplane). This means that $rank(Sim(\mathcal{C}')) \geq k-1 \geq c_{\mathbb{F}}(3)+1$.
   If $Sim(\mathcal{C}')$ were not minimal $\Rightarrow \mathcal{C}'$ is not minimal $\Rightarrow$ one of it's gates would be 0. Since $l \notin \mathcal{L}(T_0) \cup \mathcal{L}(T_1) \Rightarrow \alpha_0 T_0(v) + \alpha_1 T_1(v) \equiv 0 \Rightarrow$ for every $j \in [k]$ there exist $l_{(1-i)j} \mid T_{1-i}$ such that $l_{(1-i)j}(v), l_j(v)$ are LD. $\Rightarrow sp(\{l, l_j\}) \cap \mathcal{L}(T_{1-i}) \neq \phi$ for $j \in [k]$, a contradiction to our assumption.
   If $Sim(\mathcal{C}')$ were minimal, we have an identically zero simple minimal circuit $Sim(\mathcal{C}')$ with $rank(Sim(\mathcal{C}')) \geq c_{\mathbb{F}}(3)+1$ contradicting Theorem 1.7.
   So our assumption is wrong and $sp(\{l, l_j\}) \cap \mathcal{L}(T_{1-i}) \neq \phi$ for some $j \in [k]$.
3. Let $l \in \mathcal{C} \setminus (\mathcal{L}(T_0) \cup \mathcal{L}(T_1))$. Consider a set $\{l_1, \ldots, l_{k+2}\} \subset \mathcal{L}(T_i)$ of $k+2$ LI linear forms. By the above argument there exist three distinct elements in this set say l_1, l_2, l_3 such that $sp(\{l_j, l\}) \cap \mathcal{L}(T_{1-i}) \neq \phi$ for $j \in [3]$. Let $\{l'_1, l'_2, l'_3\} \subset \mathcal{L}(T_{1-i})$ such that $l'_j \in sp(\{l_j, l\})$ for $j \in [3]$. Then $gcd(l_j, l'_j) = 1$ implies that $l \in sp(\{l_j, l'_j\})$ for $j \in [3]$. Since l, l_j, l'_j are all standard (coefficient of x_1 is 1), Lemma 1.10 tells us

   $$l \in fl(\{l_j, l'_j\})$$

   for $j \in [3]$. So l lies on the lines $\vec{L_j} = fl(\{l_j, l'_j\})$ for $j \in [3]$. Atleast two of these lines should be distinct otherwise $dim(\{l_1, l_2, l_3\}) \leq 2$ which is a contradiction. So l is the intersection of these two lines. There are M^2 such lines and so M^4 such intersections. If $l \in \mathcal{L}(T_0) \cup \mathcal{L}(T_1)$ we have $\leq 2M$ other possibilities. So $|\mathcal{C}| \leq M^4 + 2M = O(d^4)$. ◀

   Let's now give an algorithm to construct this set.

E.2 Constructing the set $\mathcal{C}$

Here is an algorithm, Algorithm 8, to construct the set $\mathcal{C}$. An explanation is given in the lemma below.

▶ **Lemma E.3.** *Given a polynomial $f \in \mathbb{F}[x_1, \ldots, x_r]$ of degree d in r independent variables which admits a $\Sigma\Pi\Sigma_{\mathbb{F}}(2)[x_1, \ldots, x_r]$-representation : $f = \prod_{i \in [d-M]} G_i(\alpha_0 \prod_{j \in [M]} l_{0j} + \alpha_1 \prod_{k \in [M]} l_{1k})$ such that $G_t, l_{ij} (t \in [d-M], i \in \{0,1\}, j \in [M])$ are standard w.r.t. the standard basis $\{x_1, \ldots, x_n\}$ then we can find in deterministic time poly(d), the corresponding candidate set $\mathcal{C}$ (see Definition E.1) described above.*

FunctionName : Candidates
input : $f \in \Sigma\Pi\Sigma_{\mathbb{F}}(2)[\bar{x}]$
output : Set $\mathcal{C}$ of Linear Forms

1 Define $\mathcal{C} = \phi$;;
2 Use polynomial factorization from [14] to find $Lin(f)$;
3 Consider polynomial $h = \frac{f}{Lin(f)}$;
4 Let $a_2, \ldots, a_r$ be variables.;
5 Compute coefficient vector $\mathbf{b}$ of $h(a_2x_2 + \ldots + a_rx_r, x_2, \ldots, x_r)$.;
6 Consider the polynomials $\{F_i, i \in [m]\}$ constructed in Corollary A.2.;
7 Using your favorite algorithm (e.g. Buchberger's [5]) to solve polynomial equations, find all complex solutions to the system $\{F_i(\mathbf{b}) = 0, i \in [m]\}$.;
8 For each solution $(a_2, \ldots, a_r) \in \mathbb{F}^r$ do : $\mathcal{C} = \mathcal{C} \cup \{(1, a_2, \ldots, a_r)\}$;
9 **return** $\mathcal{C}$;

Algorithm 8: Set $\mathcal{C}$ of candidate linear forms

Proof. The proof also contains an explanation of the algorithm above

- Let $l = x_1 - a_2x_2 - \ldots - a_rx_r \in \mathcal{C}$ be a candidate linear form. We know that $h(a_2x_2 + \ldots + a_rx_r, x_2, \ldots, x_r) \in \Pi\Sigma_{\mathbb{F}}^{d_h}[x_2, \ldots, x_r] \subset \Pi\Sigma_{\mathbb{C}}^{d_h}[x_1, \ldots, x_r]$.
- Using Theorem A.2 we know that $h(a_2x_2 + \ldots + a_rx_r, x_2, \ldots, x_r) \in \Pi\Sigma_{\mathbb{C}}^{d_h}[x_2, \ldots, x_r] \Leftrightarrow$ for the coefficient vector $\mathbf{b}$ of $h(a_2x_2 + \ldots + a_rx_r, x_2, \ldots, x_r)$ inside $\mathbb{C}[x_2, \ldots, x_r]$ satisifes $F_1(\mathbf{b}) = \ldots = F_m(\mathbf{b}) = 0$ for the polynomials $\{F_i : i \in [m]\}$ obtained in Corollary A.2.
- For any $t \leq d_h$, computing $(a_2x_2 + \ldots + a_rx_r)^t$ requires $poly(t^r)$ time and it also has $poly(t^r)$ terms and degree t. Multiplying such powers to other variables and adding $poly(d_h^r)$ many such expressions also requires $poly(d_h^r)$ time. Hence computing the coefficient vector $\mathbf{b}$ takes polynomial time since r is a constant. Each co-ordinate of this coefficient vector is a polynomial in $r-1$ variables $(a_2, \ldots, a_r)$ of degree $poly(d_h^r)$.
- Now we think of the a_i's as our unknowns and obtain them by solving the polynomial system $\{F_i(\mathbf{b}) = 0, i \in [m]\}$. The number of polynomials is $m = poly(d^r)$ and degrees are $poly(d)$. F_i's are polynomials in $poly(d^r)$ variables. Expanding $F_i(\mathbf{b})$ will clearly take $poly(d^r)$ time and now we will have $poly(d^r)$ polynomials in r variables of degrees $poly(d^r)$. Note that $r = O(1)$ and so we need to solve $poly(d)$ polynomials of degree $poly(d)$ in constant many variables. Also Claim E.2 implies that the number of solutions $\leq M^4 + 2M = O(poly(d))$. So using Buchberger's algorithm [5] we can solve the system for $(a_2, \ldots, a_r)$ in $poly(d)$ time. Once we have the solutions we consider only those linear forms which are in $\mathbb{F}[x_1, \ldots, x_r]$ and add them to $\mathcal{C}$. ◀

F Proofs from Subsection 3.4

▶ **Claim F.1.** *Let $(S = \{l_1 \ldots, l_k\}, D)$ be a Detector pair in $\mathcal{L}(T_i)$. Let $l_{k+1} \in D$. For a standard linear form $l \in V$, if $l \mid g$ then $l \notin sp(\{l_1, \ldots, l_k\})$.*

Proof. Assume $l \mid g$ and $l \in sp(\{l_1, \ldots, l_k\})$. Let $W = sp(\{l\})$, extend it to a basis and in the process obtain W' such that $W \oplus W' = V$. We get

$$\pi_{W'}(\alpha_0 T_0 + \alpha_1 T_1) = 0\,.$$

$\pi_{W'}(\alpha_i T_i) \neq 0$ (i.e. $l \nmid T_0T_1$), otherwise l divides both T_0, T_1 and $gcd(T_0, T_1)$ won't be 1. So

we have an equality of non zero $\Pi\Sigma$ polynomials

$$\alpha_0 \prod_{j=1}^{M} \pi_{W'}(l_{0j}) = -\alpha_1 \prod_{j=1}^{M} \pi_{W'}(l_{1j}) .$$

Therefore there exists a permutation $\theta : [M] \to [M]$ such that $\{\pi_{W'}(l_{(1-i)j}), \pi_{W'}(l_{i\theta(j)})\}$ are LD $\Rightarrow l \in sp(\{l_{(1-i)j}, l_{i\theta(j)}\})$. Since $l \nmid T_0T_1$ this also means that $l_{(1-i)j} \in sp(\{l, l_{i\theta(j)}\})$ and $l_{i\theta(j)} \in sp(\{l, l_{(1-i)j}\})$.

In particular there is an $l'_{k+1} \in \mathcal{L}(T_{1-i})$ such that $l'_{k+1} \in sp(\{l, l_{k+1}\})$ and $l_{k+1} \in sp(\{l, l'_{k+1}\})$.

Since $l \in sp(\{l_1, \ldots, l_k\}) \Rightarrow l'_{k+1} \in sp(\{l_1, \ldots, l_k, l_{k+1}\})$. All linear forms here are standard(i.e. coefficient of x_1 is 1) and so by Lemma 1.10, $l'_{k+1} \in fl(\{l_1, \ldots, l_k, l_{k+1}\})$. Below we use the definition of detector pair and get

$$l'_{k+1} \in fl(\{l_1, \ldots, l_k, l_{k+1}\}) \cap \mathcal{L}(T_{1-i}) \subseteq fl(\{l_1, \ldots, l_k\}) .$$

And $l_{k+1} \in sp(\{l, l'_{k+1}\}) \Rightarrow l_{k+1} \in sp(\{l_1, \ldots, l_k\})$ which is a contradiction to (S, D) being a detector pair. ◀

▶ **Claim F.2.** *Let $l \in Lin_{\mathbb{F}}[\bar{x}]$ be standard such that $l \mid g$ and $\mathcal{C}$ be the candidate set. Assume $(S = \{l_1, \ldots, l_k\}, D(\neq \phi))$ is a Detector pair in $\mathcal{L}(T_i)$. Then $|\mathcal{L}(T_{1-i}) \cap (fl(S \cup \{l\}) \setminus fl(S))| \geq 2$. That is the flat $fl(\{l_1, \ldots, l_k, l\})$ contains atleast two distinct points from $\mathcal{L}(T_{1-i})(\subseteq \mathcal{C})$ outside $fl(\{l_1, \ldots, l_k\})$.*

Proof. From the previous claim we know that $\{l_1, \ldots, l_k, l\}$ is an LI set. Also like above we know there exists $l'_j \in \mathcal{L}(T_{1-i}), j \in [3]$ such that $l_j \in sp(\{l, l'_j\}), l'_j \in sp(\{l, l_j\})$. Since $\{l_1, l_2, l_3\}$ are LI, atleast two of the l'_j's, $j \in [3]$ must be distinct, otherwise $sp(\{l_1, l_2, l_3\}) \subset sp(\{l, l'_1\})$ which is not possible as LHS has dimension 3 and RHS has dimension 2. Thus there exist two distinct $l'_1, l'_2 \in sp(\{l_1, l_2, l_3, l\}) \subset sp(\{l_1, \ldots, l_k, l\})$. Note that $l_1, \ldots, l_k, l, l'_1, l'_2$ are all standard (i.e. coefficient of x_1 is 1) and so by Lemma 1.10

$$l'_j \in fl(\{l_1, \ldots, l_k, l\})$$

for $j \in [2]$.

If for any $j \in [2]$, $l'_j \in sp(\{l_1, \ldots, l_k\})$ then $l \in sp(\{l_j, l'_j\}) \Rightarrow l \in sp(\{l_1, \ldots, l_k\})$ which is a contradiction. This also shows that $l'_j \notin fl(\{l_1, \ldots, l_k\})$ for $j \in [2]$.

From what we showed above we may conclude:

$$l'_j \in fl(\{l_1, \ldots, l_k, l\}) \setminus fl(\{l_1, \ldots, l_k\})$$

for $j \in [2]$. Hence proved. ◀

▶ **Lemma F.3.** *The following are true:*

1. *If $l \mid I$ (i.e. l was identified) then $l \in \mathcal{L}(G) \setminus \mathcal{L}(g)$.*
2. *If $l \mid G$ (i.e. l was retained) then $(fl(\{l_1, \ldots, l_k, l\}) \setminus fl(\{l_1, \ldots, l_k\})) \cap (\mathcal{L}(T_{1-i}) \cup (\mathcal{L}(T_i) \setminus D)) \neq \phi$ that is*
   $(fl(\{l_1, \ldots, l_k, l\}) \setminus fl(\{l_1, \ldots, l_k\}))$ contains a point from $\mathcal{L}(T_i) \setminus D$ or $\mathcal{L}(T_{1-i})$.
3. *If $l \mid G$ and $l_{k+1} \in D$ then $l \notin sp(\{l_1, \ldots, l_k, l_{k+1}\})$.*

Proof.

1. Assume $l \mid I$ (i.e. l was identified) and $l \mid g$. Then by Claim 3.6 we know that $\{l_1, \ldots, l_k, l\}$ are LI and so the first "*if*" condition is true. By Claim 3.7 we know that there are two other points $\{l'_1, l'_2\} \subset \mathcal{C} \cap (fl(\{l_1, \ldots, l_k, l\}) \setminus fl(\{l_1, \ldots, l_k\}))$, so the second "*if*" condition will also be true and thus l will not be identified which is a contradiction. Therefore $l \in \mathcal{L}(G) \setminus \mathcal{L}(g)$.
2. Assume $l \mid G$ (i.e. l was not identified). This means both "*if*" statements were true for l. Thus $\{l_1, \ldots, l_k, l\}$ is LI. Also there exist distinct $\{l'_1, l'_2\} \in \mathcal{C} \cap (fl(\{l_1, \ldots, l_k, l\}) \setminus fl(\{l_1, \ldots, l_k\}))$. If

   $$l'_1 \in (\mathcal{L}(T_{1-i}) \cup (\mathcal{L}(T_i) \setminus D)) \text{ or } l'_2 \in (\mathcal{L}(T_{1-i}) \cup (\mathcal{L}(T_i) \setminus D))$$

   we are done so assume both are in

   $$\mathcal{C} \setminus ((\mathcal{L}(T_{1-i}) \cup (\mathcal{L}(T_i) \setminus D)))) = (\mathcal{C} \setminus (\mathcal{L}(T_i) \cup \mathcal{L}(T_{1-i}))) \cup D$$

   If one of them say $l'_1 \in \mathcal{C} \setminus (\mathcal{L}(T_i) \cup \mathcal{L}(T_{1-i}))$, then by Part 2 of Claim E.2, for some $j \in [k]$, $sp(\{l'_1, l_j\}) \cap \mathcal{L}(T_{1-i}) \neq \phi$. Let $\tilde{l}_j \in sp(l'_1, l_j) \cap \mathcal{L}(T_{1-i}) \Rightarrow$

   $$\tilde{l}_j \in sp(\{l'_1, l_j\}) \subseteq sp(\{l_1, \ldots, l_k, l\})$$

   Since all linear forms $\tilde{l}_j, l_1, \ldots, l_k, l$ are standard (coefficient of x_1 is 1) by Lemma 1.10

   $$\tilde{l}_j \in fl(\{l_1, \ldots, l_k, l\})$$

   Also $\tilde{l}_j, l_j$ are LI and $\tilde{l}_j \in sp(\{l'_1, l_j\})$ together imply $l'_1 \in sp(\{l_j, \tilde{l}_j\})$. Note that $l'_1 \notin fl(\{l_1, \ldots, l_k\}) \Rightarrow l'_1 \notin sp(\{l_1, \ldots, l_k\})$ which along with $l'_1 \in sp(\{l_j, \tilde{l}_j\})$ will then give

   $$\tilde{l}_j \notin sp(\{l_1, \ldots, l_k\})$$

   So we found $\tilde{l}_j \in \mathcal{L}(T_{1-i}) \cap (fl(\{l_1, \ldots, l_k, l\}) \setminus fl(\{l_1, \ldots, l_k\}))$ and we are done.

   So the only case that remains now is that $l'_1, l'_2 \in D$. Let's complete the proof in the following steps
   - $l'_1 \in fl(\{l_1, \ldots, l_k, l\}) \setminus fl(\{l_1, \ldots, l_k\}) \Rightarrow l \in sp(\{l_1, \ldots, l_k, l'_1\})$
   - Using the above bullet, $l'_2 \in fl(\{l_1, \ldots, l_k, l\}) \Rightarrow l'_2 \in sp(\{l_1, \ldots, l_k, l'_1\})$. Linear forms $l'_2, l_1, \ldots, l_k, l$ are standard (coefficient of x_1 is 1) so using Lemma 1.10, $l'_2 \in fl(\{l_1, \ldots, l_k, l'_1\})$
   - $l'_2 \in D \Rightarrow l'_2 \notin fl(\{l_1, \ldots, l_k\})$
   - The above two bullets and $\{l'_1, l'_2\} \subset \mathcal{L}(T_i)$ tell us that $fl(\{l_1, \ldots, l_k, l'_1\})$ is not elementary which is a contradiction.

   So atleast one of l'_1, l'_2 is inside $\mathcal{L}(T_{1-i}) \cup (\mathcal{L}(T_i) \setminus D)$
3. Let $l_{k+1} \in D$ and $l \in sp(\{l_1, \ldots, l_k, l_{k+1}\})$. Since $l, l_1, \ldots, l_k, l_{k+1}$ are standard, by Lemma 1.10, $l \in fl(\{l_1, \ldots, l_k, l_{k+1}\})$. Clearly $l \notin fl(\{l_1, \ldots, l_k\})$ otherwise it would get identified at the first "*if*". Therefore $l \in fl(\{l_1, \ldots, l_k, l_{k+1}\}) \setminus fl(\{l_1, \ldots, l_k\})$ By Part 2 above let $l'_1 \in (fl(\{l_1, \ldots, l_k, l\}) \setminus fl(\{l_1 \ldots, l_k\})) \cap (\mathcal{L}(T_{1-i}) \cup (\mathcal{L}(T_i) \setminus D))$. So $l'_1 \in \mathcal{L}(T_{1-i})$ or $l'_1 \in \mathcal{L}(T_i) \setminus D$.

   This tells us that $l'_1 \in sp(\{l_1, \ldots, l_k, l_{k+1}\}) \setminus fl(\{l_1, \ldots, l_k\})$. All linear forms $l'_1, l_1, \ldots, l_k, l_{k+1}$ are standard (i.e. coefficients of x_1 is 1) so by Lemma 1.10 we get that $l'_1 \in fl(\{l_1, \ldots, l_k, l_{k+1}\}) \setminus fl(\{l_1, \ldots, l_k\})$. Now using the definition of detector pair $l'_1 \notin \mathcal{L}(T_{1-i})$ since $fl(\{l_1, \ldots, l_k, l_{k+1}\}) \cap \mathcal{L}(T_{1-i}) \subseteq fl(\{l_1, \ldots, l_k\})$. The flat $fl(\{l_1, \ldots, l_k, l_{k+1}\})$ is elementary in $\mathcal{L}(T_i)$, so l'_1 can belong here only if $l'_1 = l_{k+1}$ which is not possible since $l'_1 \notin D$. So we have a contradiction. Hence proved. ◀

▶ **Lemma F.4.** *Let $(S = \{l_1, \ldots, l_k\}, D)$ be a detector in $\mathcal{L}(T_i)$. For each $(l, l_j) \in \mathcal{C} \times S$ define the space $U_{\{l,l_j\}} = sp(\{l, l_j\})$. Extend $\{l, l_j\}$ to a basis and in the process obtain $U'_{\{l,l_j\}}$ such that $V = U_{\{l,l_j\}} \oplus U'_{\{l,l_j\}}$. Define the set:*

$$X = \{l \in \mathcal{C} : \pi_{U'_{\{l,l_j\}}}(f) \neq 0, \text{ for all } l_j \in S\}$$

Then $D \subset X \subset \mathcal{L}(T_i)$.

Proof. $(D \subset X)$: Consider $l_{k+1} \in D$. Since $D \subset \mathcal{L}(T_i) \Rightarrow l_{k+1} \in \mathcal{C}$. Assume $l_{k+1} \notin X$, so there exists a $j \in [k]$ such that $\pi_{U'_{\{l_{k+1},l_j\}}}(f) = 0$. That is:

$$\pi_{U'_{\{l_{k+1},l_j\}}}(G(\alpha_0 T_0 + \alpha_1 T_1)) = 0.$$

So

$$\prod_{t\in[N_1]} \pi_{U'_{\{l_{k+1},l_j\}}}(G_t)(\alpha_0 \prod_{s\in[M]} \pi_{U'_{\{l_{k+1},l_j\}}}(l_{0s}) + \alpha_1 \prod_{s\in[M]} \pi_{U'_{\{l_{k+1},l_j\}}}(l_{1s})) = 0$$

Now

$$l_j \in \mathcal{L}(T_i) \Rightarrow \pi_{U'_{\{l_{k+1},l_j\}}}(T_i) = 0 \Rightarrow \prod_{t\in[N_1]} \pi_{U'_{\{l_{k+1},l_j\}}}(G_t) \prod_{s\in[M]} \pi_{U'_{\{l_{k+1},l_j\}}}(l_{(1-i)s}) = 0.$$

Since $G_t \mid G$, by Part (3) of Lemma 3.9 $\pi_{U'_{\{l_{k+1},l_j\}}}(G_t) \neq 0$ for all $t \in [N_1]$. If for some $s \in [M]$, $\pi_{U'_{\{l_{k+1},l_j\}}}(l_{(1-i)s}) = 0$ then $l_{(1-i)s} \in sp(\{l_j, l_{k+1}\}) \Rightarrow l_{(1-i)s} \in sp(\{l_1, \ldots, l_k, l_{k+1}\}) \Rightarrow l_{(1-i)s} \in sp(\{l_1, \ldots, l_k\})$ (by definition of Detector Pair in 3.4).

$$l_{(1-i)s} \in sp(\{l_j, l_{k+1}\}) \text{ and } \{l_{(1-i)s}, l_j\} \text{ LI } \Rightarrow l_{k+1} \in sp(\{l_{(1-i)s}, l_j\})$$

This means $l_{k+1} \in sp(\{l_1, \ldots, l_k, l_{(1-i)s}\}) \subset sp(\{l_1, \ldots, l_k\})$ which is a contradiction to $l_{k+1} \in D$. So $\pi_{U'_{\{l_{k+1},l_j\}}}(f) \neq 0$ for all $j \in [k] \Rightarrow l_{k+1} \in X$. Therefore $D \subset X$.

$(X \subset \mathcal{L}(T_i))$: Consider $l \in X$. We need to show $l \in \mathcal{L}(T_i)$. We already know $l \in \mathcal{C}$.

- If $l \in \mathcal{L}(T_{1-i})$, then $\pi_{U'_{\{l,l_j\}}}(f) = 0$ for all $j \in [k]$ since $l \mid T_{1-i}$ and $l_j \mid T_i$. Contradiction to $l \in X$.
- If $l \in \mathcal{C} \setminus (\mathcal{L}(T_i) \cup \mathcal{L}(T_{1-i}))$ by Part 2 of Claim E.2 we know that there exists $j \in [k]$ such that $sp(\{l_j, l\}) \cap \mathcal{L}(T_{1-i}) \neq \phi$. Let $l'_j \in sp(\{l_j, l\}) \cap \mathcal{L}(T_{1-i})$. We show that $sp(\{l'_j, l_j\}) = sp(\{l_j, l\}) = U_{\{l_j,l\}}$.
  - $l'_j \in sp(\{l_j, l\}) \Rightarrow sp(\{l'_j, l_j\}) \subset sp(\{l_j, l\})$.
  - Let $l'_j = \alpha l_j + \beta l$. We know that $\{l_j, l'_j\}$ are LI since $l_j \in \mathcal{L}(T_i)$ and $l'_j \in \mathcal{L}(T_{1-i})$. So $\beta \neq 0 \Rightarrow l \in sp(\{l'_j, l_j\}) \Rightarrow sp(\{l, l_j\}) \subset sp(\{l'_j, l_j\}) \Rightarrow sp(\{l, l_j\}) = sp(\{l'_j, l_j\})$.

  Use the same extension for $sp(\{l, l_j\}) = sp(\{l'_j, l_j\}) = U_{\{l_j,l\}}$ to get $\pi_{U'_{\{l,l_j\}}}(f) = \pi_{U'_{\{l'_j,l_j\}}}(f) = 0$ (since $l'_j \mid T_{1-i}$ and $l_j \mid T_i$). Contradiction to $l \in X$.

Therefore $l \in \mathcal{L}(T_i) \Rightarrow X \subset \mathcal{L}(T_i)$. ◀

G Proofs from Subsection 3.5

▶ **Claim G.1.** *The following is true*

$$\frac{(2 - v(\delta, \theta))}{v(\delta, \theta)} \leq \frac{1 - \delta}{\delta}.$$

Proof. Note that

$$\frac{(2-v(\delta,\theta))}{v(\delta,\theta)} = \begin{cases} \frac{1+\delta+\theta}{1-\delta-\theta} & \text{if } |\mathcal{L}(T_0)| \leq \theta|\mathcal{L}(T_1)| \\ \frac{3-(1-\delta)(1+\theta)}{(1-\delta)(1+\theta)-1} & \text{if } \theta|\mathcal{L}(T_1)| < |\mathcal{L}(T_0)| \leq |\mathcal{L}(T_1)| \end{cases}$$

By simple computation $\delta \in (0, \frac{7-\sqrt{37}}{6})$ gives

$$3\delta^2 - 7\delta + 1 > 0 \Rightarrow 0 < \frac{3\delta}{1-\delta} < 1 - 3\delta < 1 \Rightarrow \frac{1+\delta+\theta}{1-\delta-\theta} < \frac{1-\delta}{\delta}$$

Also

$$\theta > \frac{3\delta}{1-\delta} \Rightarrow \frac{3-(1-\delta)(1+\theta)}{(1-\delta)(1+\theta)-1} < \frac{1-\delta}{\delta}$$

◀

▶ **Lemma G.2.** *Let $k = c_{\mathbb{F}}(3) + 2$ (see definition of $c_{\mathbb{F}}(k)$ in Theorem 1.7). Fix δ, θ in range given in Claim 3.12 above. Then for some $i \in \{0,1\}$ there exists a Detector Pair $(S = \{l_1, \ldots, l_k\}, D)$ in $\mathcal{L}(T_i)$ with $|D| \geq v(\delta,\theta)\max(|\mathcal{L}(T_0)|, |\mathcal{L}(T_1)|)$.*

Proof. We assume $|\mathcal{L}(T_0)| \leq \mathcal{L}(T_1)$. The other case gives the same result for(maybe) a different value of i. We will consider linear forms as points in the space $\mathbb{F}^r$. Let's consider the two cases used in the definition of $v(\delta,\theta)$.

Case 1: $|\mathcal{L}(T_0)| \leq \theta|\mathcal{L}(T_1)|$ (i.e. $\mathcal{L}(T_0)$ is much smaller) $\Rightarrow v(\delta,\theta) = 1-\delta-\theta$ Since $dim(\mathcal{L}(T_1)) \geq r-1 \geq C_{2k-1} > C_k$ (see Appendix B for definition of C_k) by Corollary B.5 there exists a set S of k LI points say $S = \{l_1, \ldots, l_k\} \subseteq \mathcal{L}(T_1)$ and a set $Z \subseteq \mathcal{L}(T_1)$ of size $\geq (1-\delta)|\mathcal{L}(T_1)|$ such that for any $l_{k+1} \in Z$

- $l_{k+1} \notin fl(\{l_1, \ldots, l_k\})$.
- $fl(\{l_1, \ldots, l_k, l_{k+1}\})$ is elementary in $\mathcal{L}(T_1)$.

Next we define our set D according to the condition we needed in the definition of detector (See Subsection 3.4).

$$D \stackrel{def}{=} \{l_{k+1} \in Z : fl(\{l_1, \ldots, l_k, l_{k+1}\}) \cap \mathcal{L}(T_0) \subset fl(\{l_1, \ldots, l_k\})\}$$

In the following lines we will show that this set D has large size, to be precise:

$$|D| \geq (1-\delta-\theta)|\mathcal{L}(T_1)|$$

We do this in steps:

1. First we define a special subset of Z

   $$\tilde{Z} = \{l_{k+1} \in Z : (fl(\{l_1, \ldots, l_{k+1}\}) \setminus fl(\{l_1, \ldots, l_k\})) \cap \mathcal{L}(T_0) \neq \phi\}$$

   We claim that $Z \setminus \tilde{Z} \subset D$. Let $l_{k+1} \in Z \setminus \tilde{Z} \Rightarrow (fl(\{l_1, \ldots, l_{k+1}\}) \setminus fl(\{l_1, \ldots, l_k\})) \cap \mathcal{L}(T_0) = \phi \Rightarrow fl(\{l_1, \ldots, l_{k+1}\}) \cap \mathcal{L}(T_0) \subset fl(\{l_1, \ldots, l_k\})$ and so $l_{k+1} \in D$.
2. Next we show that for distinct $l_{k+1}, \tilde{l}_{k+1} \in Z(\subseteq \mathcal{L}(T_1))$

   $$(fl(\{l_1, \ldots, l_k, l_{k+1}\}) \setminus fl(\{l_1, \ldots, l_k\})) \cap (fl(\{l_1, \ldots, l_k, \tilde{l}_{k+1}\}) \setminus fl(\{l_1, \ldots, l_k\})) = \phi$$

   If not then there exist scalars $\mu_j, \nu_j, j \in [k+1]$ such that

   $$\nu_1 l_1 + \ldots \nu_k l_k + \nu_{k+1} l_{k+1} = \mu_1 l_1 + \ldots \mu_k l_k + \mu_{k+1} \tilde{l}_{k+1}$$

   with $\nu_{k+1} \neq 0$ implying that $l_{k+1} \in sp(\{l_1, \ldots, l_k, \tilde{l}_{k+1}\})$. Since all linear forms are *standard* this implies $l_{k+1} \in fl(\{l_1, \ldots . l_k, \tilde{l}_{k+1}\})$ (see Lemma 1.10). Also $l_{k+1} \in Z \Rightarrow l_{k+1} \notin fl(\{l_1, \ldots, l_k\})$. Together this means that $l_{k+1} \in fl(\{l_1, \ldots, l_k, \tilde{l}_{k+1}\}) \setminus fl(l_1, \ldots, l_k)$ and we arrive at a contradiction to $fl(\{l_1, \ldots, l_k, \tilde{l}_{k+1}\})$ being elementary.

3. From what we showed above every $l \in \mathcal{L}(T_0)$ can belong to atmost one of the sets $fl(\{l_1,\ldots,l_{k+1}\}) \setminus fl(\{l_1,\ldots,l_k\})$ with $l_{k+1} \in Z$ (since intersection between two such sets is ϕ) and therefore there can be atmost $|\mathcal{L}(T_0)|$ such l_{k+1}'s in $\tilde{Z} \Rightarrow |\tilde{Z}| \leq |\mathcal{L}(T_0)|$.

   So we get:

   $$|D| \geq |Z| - |\mathcal{L}(T_0)| \geq (1-\delta-\theta)|\mathcal{L}(T_1)|$$

(S, D) is a detector pair in $\mathcal{L}(T_1)$ by the choice of Z and D.

Case 2: $\theta|\mathcal{L}(T_1)| < |\mathcal{L}(T_0)| \leq |\mathcal{L}(T_1)|$ (i.e. sizes are comparable) $\Rightarrow v(\delta,\theta) = (1-\delta)(1+\theta)-1$ Since $dim(\mathcal{L}(T_0) \cup \mathcal{L}(T_1)) = r > C_{2k-1}$, by Corollary B.5 we know that there exist $2k-1$ independent points $l_1,\ldots,l_{2k-1} \in \mathcal{L}(T_0) \cup \mathcal{L}(T_1)$ and a set $Z \subseteq \mathcal{L}(T_0) \cup \mathcal{L}(T_1)$ of size $\geq (1-\delta)(|\mathcal{L}(T_0)| + |\mathcal{L}(T_1)|)$ such that for all $l \in Z$

- $l \notin fl(\{l_1,\ldots,l_{2k-1}\})$.
- $fl(\{l_1,\ldots,l_{2k-1},l\})$ is elementary in $\mathcal{L}(T_0) \cup \mathcal{L}(T_1)$.

By pigeonhole principle, k of the $\{l_j\}_{j=1}^{2k-1}$ points must belong to either $\mathcal{L}(T_0)$ or $\mathcal{L}(T_1)$. Let's assume they belong to $\mathcal{L}(T_i)$ (for some $i \in \{0,1\}$) (say the points are $l_1,\ldots,l_k$), then consider $D = Z \cap \mathcal{L}(T_i)$. Clearly for every $l \in D$, $l \notin fl(\{l_1,\ldots,l_k\})$ and $fl(\{l_1,\ldots,l_k,l\})$ is elementary in $\mathcal{L}(T_0) \cup \mathcal{L}(T_1)$. This immediately tells us that $(S = \{l_1,\ldots,l_k\}, D)$ satisfies all properties of being a detector pair in $\mathcal{L}(T_i)$. We defined $D = Z \cap \mathcal{L}(T_i)$. Since $Z \subseteq \mathcal{L}(T_i) \cup \mathcal{L}(T_{1-i})$ we have $Z = (Z \cap \mathcal{L}(T_i)) \cup (Z \cap \mathcal{L}(T_{1-i})) \subset D \cup \mathcal{L}(T_{1-i})$ giving

$$|D| + |\mathcal{L}(T_{1-i})| \geq |Z| \Rightarrow |D| \geq |Z| - |\mathcal{L}(T_{1-i})| \geq (1-\delta)(|\mathcal{L}(T_0)| + |\mathcal{L}(T_1)|) - |\mathcal{L}(T_{1-i})|$$

$$\geq ((1-\delta)(1+\theta)-1)\max(|\mathcal{L}(T_0)|, |\mathcal{L}(T_1)|)$$

Combining the two cases we see that for some $i \in \{0,1\}$ there exists a Detector set $(S = \{l_1,\ldots,l_k\}, D)$ in $\mathcal{L}(T_i)$ with $|D| \geq v(\delta,\theta)\max(|\mathcal{L}(T_0)|, |\mathcal{L}(T_1)|)$. ◀

▶ **Lemma G.3.** *The following are true:*

1. $dim(\pi_{W_0^\perp}(\mathcal{L}(U_{1-i}))) > C_4$
2. $\pi_{W_0^\perp}(\mathcal{L}(U_{1-i})) \cap \pi_{W_0^\perp}(D) = \phi$
3. $|\pi_{W_0^\perp}(\mathcal{L}(U_{1-i}))| \leq \frac{1-\delta}{\delta}|\pi_{W_0^\perp}(D)|$

Proof.

1. Since $dim(\mathcal{L}(U_{1-i})) \geq r-1$ we get $dim(\pi_{W_0^\perp}(\mathcal{L}(U_{1-i}))) \geq r-1-k > C_4$.
2. Assume $\exists\ d_1 \in D, u \in \mathcal{L}(U_{1-i})$ such that $\pi_{W_0^\perp}(d) = \pi_{W_0^\perp}(u) \Rightarrow \exists \lambda,\nu \in \mathbb{F}$ such that $\nu d_1 + \lambda u \in W_0^\perp$. Since $\pi_{\bar{W}_0}(d_1) \neq 0$ both $\nu,\lambda \neq 0$. Thus $u \in sp(\{l_1,\ldots,l_k,d_1\}) \Rightarrow u \in fl(\{l_1,\ldots,l_k,d_1\})$ (using Lemma 1.10 since all linear forms involved are *standard* i.e. have coefficient of x_1 equal to 1). Also $u \in \mathcal{L}(GT_{1-i}) \Rightarrow u \in fl(\{l_1,\ldots,l_k,d_1\}) \cap (\mathcal{L}(G) \cup \mathcal{L}(T_{1-i}))$. We know from Part (2) of Lemma 3.9 that $fl(\{l_1,\ldots,l_k,d_1\}) \cap \mathcal{L}(G) = \phi \Rightarrow u \in fl(\{l_1,\ldots,l_k,d_1\}) \cap \mathcal{L}(T_{1-i}) \subseteq fl\{l_1,\ldots,l_k\}$ because (S,D) was a detector pair. But $u \in fl(\{l_1,\ldots,l_k\}) \Rightarrow d_1 \in sp(\{l_1,\ldots,l_k\})$ which is a contradiction because $d_1 \in D$ and (S,D) is a detector pair.
3. We first plan to show $\pi_{W_0^\perp}(\mathcal{L}(U_{1-i})) \subset \pi_{W_0^\perp}(\mathcal{L}(T_{1-i})) \cup \pi_{W_0^\perp}(\mathcal{L}(T_i) \setminus D)$. Clearly $U_{1-i} \mid GT_{1-i} \Rightarrow \mathcal{L}(U_{1-i}) \subset \mathcal{L}(GT_{1-i}) \Rightarrow \pi_{W_0^\perp}(\mathcal{L}(U_{1-i})) \subset \pi_{W_0^\perp}(\mathcal{L}(GT_{1-i})) \subset \pi_{W_0^\perp}(\mathcal{L}(G)) \cup \pi_{W_0^\perp}(\mathcal{L}(T_{1-i}))$. Now consider any $l \in \mathcal{L}(G)$. We know that $(S_0 = \{l_1,\ldots,l_k\}, D)$ is a detector pair, so by Part (2) of Lemma 3.9 we get

   $$(fl(\{l_1,\ldots,l_k,l\}) \setminus fl(\{l_1,\ldots,l_k\})) \cap (\mathcal{L}(T_{1-i}) \cup (\mathcal{L}(T_i) \setminus D)) \neq \phi$$

So there exists $l' \in \mathcal{L}(T_{1-i}) \cup (\mathcal{L}(T_i) \setminus D)$ such that $\pi_{W_0^\perp}(l), \pi_{W_0^\perp}(l')$ are both non-zero and are LD $\Rightarrow \pi_{W_0^\perp}(l) = \pi_{W_0^\perp}(l')$ implying that $\pi_{W_0^\perp}(\mathcal{L}(G)) \subset \pi_{W_0^\perp}(\mathcal{L}(T_{1-i}) \cup (\mathcal{L}(T_i) \setminus D))$ giving us $\pi_{W_0^\perp}(\mathcal{L}(U_{1-i})) \subset \pi_{W_0^\perp}(\mathcal{L}(T_{1-i})) \cup \pi_{W_0^\perp}(\mathcal{L}(T_i) \setminus D)$ and therefore

$$|\pi_{W_0^\perp}(\mathcal{L}(U_{1-i}))| \leq |\pi_{W_0^\perp}(\mathcal{L}(T_{1-i}))| + |\pi_{W_0^\perp}(\mathcal{L}(T_i) \setminus D)|$$

Now we try to show $|\pi_{W_0^\perp}(\mathcal{L}(T_i) \setminus D)| = |\pi_{W_0^\perp}(\mathcal{L}(T_i))| - |D|$

a. It's straightforward to see $\pi_{W_0^\perp}(\mathcal{L}(T_i)) = \pi_{W_0^\perp}(D) \cup \pi_{W_0^\perp}(\mathcal{L}(T_i) \setminus D)$. Also $\pi_{W_0^\perp}(\mathcal{L}(T_i) \setminus D) \cap \pi_{W_0^\perp}(D) = \phi$. If not then there exists $l' \in \mathcal{L}(T_i) \setminus D, l'' \in D$ such that $0 \neq \pi_{W_0^\perp}(l'') = \pi_{W_0^\perp}(l') \Rightarrow \pi_{W_0^\perp}(l''), \pi_{W_0^\perp}(l')$ are LD $\Rightarrow l' \in sp\{l_1, \ldots, l_k, l''\} \setminus sp\{l_1, \ldots, l_k\} \Rightarrow$ (by Lemma 1.10), $l' \in fl\{l_1, \ldots, l_k, l''\} \setminus fl\{l_1, \ldots, l_k\}$ which is a contradiction to the flat being elementary inside $\mathcal{L}(T_i)$. So $|\pi_{W_0^\perp}(\mathcal{L}(T_i))| = |\pi_{W_0^\perp}(D)| + |\pi_{W_0^\perp}(\mathcal{L}(T_i) \setminus D)|$.

b. $\pi_{W_0^\perp}$ is injective on D. Let $\pi_{W_0^\perp}(l') = \pi_{W_0^\perp}(l'')$ for LI forms $\{l', l''\} \subset D$, then $l' \in sp(\{l_1, \ldots, l_k, l''\}) \Rightarrow$ (by Lemma 1.10), $l' \in fl(\{l_1, \ldots, l_k, l''\})$ and clearly $l' \notin fl\{l_1, \ldots, l_k\}$ (since it's in D), which is again a contradiction to the flat being elementary , thus $|\pi_{W_0^\perp}(D)| = |D| = |D|$ (since D is a set of *normal* linear forms).

Combining these with Claim 3.12 and Lemma 3.13 we get

$$|\pi_{W_0^\perp}(\mathcal{L}(U_{1-i}))| \leq 2\max(|\mathcal{L}(T_0)|, |\mathcal{L}(T_1)|) - |D| \leq (2 - v(\delta, \theta)) \max(|\mathcal{L}(T_0)|, |\mathcal{L}(T_1)|)$$

$\Rightarrow$

$$\frac{|\pi_{W_0^\perp}(\mathcal{L}(U_{1-i}))|}{|\pi_{W_0^\perp}(D)|} \leq \frac{(2 - v(\delta, \theta))}{v(\delta, \theta)} \leq \frac{1 - \delta}{\delta}$$

◀

H Proofs from Section 4

Our field $\mathbb{F}$ has characteristic zero. For simplicity let's assume it is an extension of $\mathbb{Q}$ and therefore contains $\mathbb{Z}$. All random selections are done from the set $[N] = \{1, \ldots, N\}$.

▶ **Lemma H.1.** *Let $\mathbb{F}^n$ be the n dimensional vector space over $\mathbb{F}$. Suppose $v_i : i \in [n]$ are vectors in $\mathbb{F}^n$ with each co-ordinate chosen independently from the uniform distribution on $[N]$. Consider the event*

$$\mathcal{E} = \{\{v_1, \ldots, v_n\} \text{ are LI }\}.$$

Then $Pr[\mathcal{E}] \geq 1 - \frac{n}{N}$.

Proof. Each $v_i \in \mathbb{F}^n$ is chosen such that each co-ordinate is chosen uniformly randomly from the set $[N]$. Let v_i be the vector $(V_{i,1}, \ldots, V_{i,n})$. Consider the matrix $\tilde{V} = (V_{i,j})$. The v_i's will be linearly independent if and only if $\tilde{V}$ is invertible i.e. $det(V_{i,j}) \neq 0$. Note that $det(V_{i,j})$ is not the zero polynomial since the monomial $V_{1,1}V_{2,2}\ldots V_{n,n}$ has coefficient 1. Now we can use Schwartz-Zippel Lemma [21] on this polynomial to yield:

$$Pr[det(\tilde{V}) = 0]. \leq \frac{n}{N}$$

Therefore $Pr[\mathcal{E}] = Pr[det(\tilde{V}) \neq 0] \geq 1 - \frac{n}{N}$. ◀

▶ **Lemma H.2.** *Assume conditions in the previous lemma. For a fixed r, consider the subspaces $V = sp\{v_1, \ldots, v_r\}$ and $V' = sp\{v_{r+1}, \ldots, v_n\}$. Let's assume that that $\mathcal{E}$ occurs i.e. $\{v_1, \ldots, v_n\}$ are LI. So $dim(V) = r$. We know $\mathbb{F}^n = V \oplus V'$. Let $\pi_V : \mathbb{F}^n \to V$ be the*

orthogonal projection onto V under this decomposition. Let $T \subset \mathbb{F}^n$ be finite. Consider the event

$$\mathcal{F} = \{\exists \text{ an LI set } \{l_1, \ldots, l_r\} \subset T \text{ such that } \{\pi_V(l_1), \ldots, \pi_V(l_r)\} \text{ is LD } \}.$$

Then $Pr[\mathcal{F}] \leq \binom{|T|}{r}\{\frac{n}{N} + \frac{r(n-1)}{N}\}$.

Proof. Fix $\{l_1, \ldots, l_r\} \subset T$ an LI set. Extend it to get a basis $\{l_1, \ldots, l_n\}$ of $\mathbb{F}^n$. Let $l_i = \sum_{j \in [n]} L_{i,j} e_j$ and L be the matrix $(L_{i,j})$. From the discussion above we have $\tilde{V} = (V_{i,j})$. Now let P_r be the $n \times n$ matrix

$$P_r = \begin{bmatrix} I_r & 0_{r,n-r} \\ 0_{n-r,r} & 0_{n-r,n-r} \end{bmatrix}$$

where I_r is the $r \times r$ identity matrix and $0_{p,q}$ is the $p \times q$ matrix with all 0 entries. Also for any $n \times n$ matrix A, define $M_r(A)$ to be the principal $r \times r$ minor of A. Consider the equation given by

$$det(M_r(P_r Lco(\tilde{V}))) = 0$$

where $co(\tilde{V})$ is the co-factor matrix of $\tilde{V}$. Since entries of $co(\tilde{V})$ are polynomials in the $V_{i,j}$'s and L is a fixed matrix, the entries of $P_r Lco(\tilde{V})$ are polynomials in $V_{i,j}$'s. So $det(M_r(P_r Lco(\tilde{V})))$ is a polynomial in $V_{i,j}$'s. This polynomial can't be identically 0. Choose $V_{i,j} = L_{i,j}$, then since $\tilde{V}$ is invertible, $Lco(\tilde{V}) = det(L)I$ giving $P_r Lco(\tilde{V}) = det(L)P_r \Rightarrow det(M_r(P_r Lco(\tilde{V}))) = det(L) \neq 0$. Degree of the polynomial $det(M_r(P_r Lco(\tilde{V})))$ is clearly $\leq r(n-1)$. Therefore by Schwartz Zippel Lemma

$$Pr[det(M_r(P_r Lco(\tilde{V}))) = 0] \leq \frac{r(n-1)}{N}.$$

Consider the set

$$S(\{l_1, \ldots, l_r\}) = \{(V_{i,j}) : det(\tilde{V}) \neq 0, det(M_r(P_r Lco(\tilde{V})) \neq 0\}.$$

On this set $S(\{l_1, \ldots, l_r\})$, $\{v_1, \ldots, v_n\}$ is a basis and we have the following matrix equations:

$$\begin{bmatrix} v_1 \\ . \\ . \\ v_n \end{bmatrix} = \tilde{V} \begin{bmatrix} e_1 \\ . \\ . \\ e_n \end{bmatrix} \text{ and } \begin{bmatrix} l_1 \\ . \\ . \\ l_n \end{bmatrix} = L \begin{bmatrix} e_1 \\ . \\ . \\ e_n \end{bmatrix} \Rightarrow \begin{bmatrix} l_1 \\ . \\ . \\ l_n \end{bmatrix} = L\tilde{V}^{-1} \begin{bmatrix} v_1 \\ . \\ . \\ v_n \end{bmatrix}$$

and so

$$\begin{bmatrix} \pi_V(l_1) \\ . \\ \pi_V(l_r) \end{bmatrix} = \frac{1}{det(\tilde{V})} M_r(P_r Lco(\tilde{V})) \begin{bmatrix} v_1 \\ . \\ v_r \end{bmatrix}$$

Therefore $\{\pi_V(l_1), \ldots, \pi_V(l_r)\}$ is an LI set. Now $S(\{l_1, \ldots, l_r\})^c = \{(V_{i,j}) : det(\tilde{V}) = 0$ or $det(M_r Lco(M)) = 0\} \Rightarrow Pr[S(\{l_1, \ldots, l_r\})^c] \leq \frac{n}{N} + \frac{r(n-1)}{N}$. Next we vary $\{l_1, \ldots, l_r\}$ and apply union bound to get

$$Pr[\mathcal{F}] \leq \sum_{\{l_1, \ldots, l_r\} \subset T} S(\{l_1, \ldots, l_r\})^c \leq \binom{|T|}{r} \left\{ \frac{n}{N} + \frac{r(n-1)}{N} \right\}.$$

In our application $|T| = poly(d)$ and r is a constant, so we choose $N = 2^{d+n}$ and make this probability very small. ◀

▶ **Lemma H.3.** *Let* $f|_V(\bar{X}) = \sum_{\{\bar{\alpha}:|\bar{\alpha}|=d\}} a_{\bar{\alpha}}\bar{X}^{\bar{\alpha}}$ *be a homogeneous multivariate polynomial of degree* d *in* r *variables* $X_1, \ldots, X_r$. *Let* $p_i : 1 \leq i \leq \binom{d+r-1}{r-1}$ *be randomly chosen points in* V *(dimension* r *random subspace of* $\mathbb{F}^n$ *chosen in the above lemmas). Then with high probability one can find all the* $a_{\bar{\alpha}}$.

Proof. We evaluate the polynomial at each of the p_i's. So we have $\binom{d+r-1}{r-1}$ evaluations. The number of coefficients is also $\binom{d+r-1}{r-1}$ so we get a linear system in the coefficients where the matrix (X) entries are just monomials evaluated at the p_i's. Since f is not identically zero clearly there exist values for the points p_i's such that the determinant of this matrix is non zero polynomial so it cannot be identically zero. Now the degree of the determinant polynomial is bounded by $d\binom{d+r-1}{r-1} \leq poly((d+r)^r)$. So by Schwarz Zippel lemma

$$Pr[a_{\bar{\alpha}} \text{ is recovered correctly }] = Pr[det(X) \neq 0] \geq 1 - \frac{poly(d^r)}{N}$$ ◀

Proof Complexity Lower Bounds from Algebraic Circuit Complexity*

Michael A. Forbes[1], Amir Shpilka[2], Iddo Tzameret[3], and Avi Wigderson[4]

1 **Department of Computer Science, Princeton University, Princeton, USA**
miforbes@csail.mit.edu
2 **Department of Computer Science, Tel Aviv University, Tel Aviv, Israel**
shpilka@post.tau.ac.il
3 **Department of Computer Science, Royal Holloway, University of London, Egham, UK**
iddo.tzameret@rhul.ac.uk
4 **School of Mathematics, Institute for Advanced Study, Princeton, USA**
avi@math.ias.edu

Abstract

We give upper and lower bounds on the power of subsystems of the Ideal Proof System (IPS), the algebraic proof system recently proposed by Grochow and Pitassi [26], where the circuits comprising the proof come from various restricted algebraic circuit classes. This mimics an established research direction in the boolean setting for subsystems of Extended Frege proofs whose lines are circuits from restricted boolean circuit classes. Essentially all of the subsystems considered in this paper can simulate the well-studied Nullstellensatz proof system, and prior to this work there were no known lower bounds when measuring proof size by the algebraic complexity of the polynomials (except with respect to degree, or to sparsity).

Our main contributions are two general methods of converting certain algebraic lower bounds into proof complexity ones. Both require stronger arithmetic lower bounds than common, which should hold not for a specific polynomial but for a whole family defined by it. These may be likened to some of the methods by which Boolean circuit lower bounds are turned into related proof-complexity ones, especially the "feasible interpolation" technique. We establish algebraic lower bounds of these forms for several explicit polynomials, against a variety of classes, and infer the relevant proof complexity bounds. These yield separations between IPS subsystems, which we complement by simulations to create a partial structure theory for IPS systems.

Our first method is a *functional lower bound*, a notion of Grigoriev and Razborov [25], which is a function $\hat{f} : \{0,1\}^n \to \mathbb{F}$ such that any polynomial f agreeing with $\hat{f}$ on the boolean cube requires large algebraic circuit complexity. We develop functional lower bounds for a variety of circuit classes (sparse polynomials, depth-3 powering formulas, read-once algebraic branching programs and multilinear formulas) where $\hat{f}(\vec{x})$ equals $1/p(\vec{x})$ for a constant-degree polynomial p depending on the relevant circuit class. We believe these lower bounds are of independent interest in algebraic complexity, and show that they also imply lower bounds for the size of the corresponding IPS refutations for proving that the relevant polynomial p is non-zero over the boolean cube. In particular, we show super-polynomial lower bounds for refuting variants of the subset-sum axioms in these IPS subsystems.

Our second method is to give *lower bounds for multiples*, that is, to give explicit polynomials whose all (non-zero) multiples require large algebraic circuit complexity. By extending known techniques, we give lower bounds for multiples for various restricted circuit classes such

* The research leading to these results has received funding from the Princeton Center for Theoretical Computer Science, the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement number 257575, and NSF grant CCF-1412958.

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 32; pp. 32:1–32:17

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

sparse polynomials, sums of powers of low-degree polynomials, and roABPs. These results are of independent interest, as we argue that lower bounds for multiples is the correct notion for instantiating the algebraic hardness versus randomness paradigm of Kabanets and Impagliazzo [31]. Further, we show how such lower bounds for multiples extend to lower bounds for refutations in the corresponding IPS subsystem.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems – Complexity of Proof Procedures, F.2.1 Numerical Algorithms and Problems, I.1.1 Expressions and their Representation

Keywords and phrases Proof Complexity, Algebraic Complexity, Nullstellensatz, Subset-Sum

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.32

1 Introduction

Propositional proof complexity aims to understand and analyze the computational resources required to prove propositional tautologies, in the same way that circuit complexity studies the resources required to compute boolean functions. A typical goal would be to establish, for a given proof system, super-polynomial lower bounds on the *size* of any proof of some propositional tautology. The seminal work of Cook and Reckhow [13] showed that this goal relates quite directly to fundamental hardness questions in computational complexity such as the NP vs. coNP question: establishing super-polynomial lower bounds for *every* propositional proof system would separate NP from coNP (and thus also P from NP). We refer the reader to Krajíček [35] for more on this subject.

Propositional proof systems come in a large variety, as different ones capture different forms of reasoning, either reasoning used to actually prove theorems, or reasoning used by algorithmic techniques for different types of search problems (as failure of the algorithm to find the desired object constitutes a proof of its nonexistence). Much of the research in proof complexity deals with propositional proof systems originating from logic and from geometry. Logical proof systems include such systems as *resolution* (whose variants are related to popular algorithms for automated theory proving and SAT solving), as well as the *Frege* proof system (capturing the most common logic text-book systems) and its many subsystems. Geometric proof systems include *cutting-plane proofs*, capturing reasoning used in algorithms for integer programming, as well as proof systems arising from systematic strategies for rounding linear- or semidefinite-programming such as the lift-and-project or sum-of-squares hierarchies.

In this paper we focus on algebraic proof systems, in which propositional tautologies (or rather contradictions) are expressed as unsatisfiable systems of polynomial equations and algebraic tools are used to refute them. This study originates with the work of Beame, Impagliazzo, Krajíček, Pitassi and Pudlák [6], who introduced the Nullstellensatz refutation system (based on Hilbert's Nullstellensatz), followed by the Polynomial Calculus system of Clegg-Edmonds-Impagliazzo [10], which is a "dynamic version" of Nullstellensatz. In both systems the main measures of proof size that have been studied are the *degree* and *sparsity* of the polynomials appearing in the proof. Substantial work has lead to a very good understanding of the power of these systems with respect to these measures (see for example [9, 50, 23, 30, 8, 4] and references therein).

However, the above measures of degree and sparsity are rather rough measures of a complexity of a proof. As such, Grochow and Pitassi [26] have recently advocated measuring

the complexity of such proofs by their algebraic circuit size and shown that the resulting proof system can polynomially simulate strong proof systems such as the Frege system. This naturally leads to the question of establishing lower bounds for this stronger proof system, even for restricted classes of algebraic circuits.

In this work we establish such lower bounds for previously studied restricted classes of algebraic circuits, and show these lower bounds are interesting by providing non-trivial *upper* bounds in these proof systems for refutations of interesting sets of polynomial equations. This provides what are apparently the first examples of lower bounds on the algebraic circuit size of propositional proofs in the ideal proof system (IPS) framework of Grochow and Pitassi [26].

We note that obtaining proof complexity lower bounds from circuit complexity lower bounds is an established tradition, and takes many forms. Most prominent are the lower bounds for susbsystems of the Frege proof system defined by low-depth Boolean circuits, and lower bounds on Resolution and Cutting Planes system using the so-called feasible interpolation method [44]. We refer the reader again to the monograph [35] for more details. Our approach here for algebraic systems shares features with both of these approaches.

The rest of this long introduction is arranged as follows. In Subsection 1.1 we give the necessary background in algebraic proof complexity, and explain the IPS system. In subsection 1.2 we define the algebraic complexity classes that will underlie the subsystems of IPS we will study. In subsection 1.3 we state our results and explain our techniques, for both the algebraic and proof complexity worlds.

2 Algebraic Proof Systems

We now describe the algebraic proof systems that are the subject of this paper. If one has a set of polynomials (called *axioms*) $f_1, \ldots, f_m \in \mathbb{F}[x_1, \ldots, x_n]$ over some field $\mathbb{F}$, then (the weak version of) Hilbert's Nullstellensatz shows that the system $f_1(\vec{x}) = \cdots = f_m(\vec{x}) = 0$ is unsatisfiable (over the algebraic closure of $\mathbb{F}$) if and only if there are polynomials $g_1, \ldots, g_m \in \mathbb{F}[\vec{x}]$ such that $\sum_j g_j(\vec{x}) f_j(\vec{x}) = 1$ (as a formal identity), or equivalently, that 1 is in the ideal generated by the $\{f_j\}_j$.

Beame, Impagliazzo, Krajíček, Pitassi, and Pudlák [6] suggested to treat these $\{g_j\}_j$ as a *proof* of the unsatisfiability of this system of equations, called a *Nullstellensatz refutation*. This is particular relevant for complexity theory as one can restrict attention to *boolean* solutions to this system by adding the *boolean axioms*, that is, adding the polynomials $\{x_i^2 - x_i\}_{i=1}^n$ to the system. As such, one can then naturally encode NP-complete problems such as the satisfiability of 3CNF formulas as the satisfiability of a system of constant-degree polynomials, and a Nullstellensatz refutation is then an equation of the form $\sum_{j=1}^m g_j(\vec{x}) f_j(\vec{x}) + \sum_{i=1}^n h_i(\vec{x})(x_i^2 - x_i) = 1$ for $g_j, h_i \in \mathbb{F}[\vec{x}]$. This proof system is sound (only refuting unsatisfiable systems over $\{0,1\}^n$) and complete (refuting any unsatisfiable system, by Hilbert's Nullstellensatz).

Given that the above proof system is sound and complete, it is then natural to ask what is its power to refute unsatisfiable systems of polynomial equations over $\{0,1\}^n$. To understand this question one must define the notion of the *size* of the above refutations. Two popular notions are that of the *degree*, and the *sparsity* (number of monomials). One can then show (see for example Pitassi [43]) that for any unsatisfiable system which includes the boolean axioms, there exist a refutation where the g_j are multilinear and where the h_i have degree at most $O(n + d)$, where each f_j has degree at most d. In particular, this implies, when $d = O(n)$, that for any unsatisfiable system there is a refutation of degree $O(n)$ and involving at most $\exp(O(n))$ monomials. This intuitively agrees with the fact that coNP is a subset of non-deterministic exponential time.

Building on the suggestion of Pitassi [43], Grochow and Pitassi [26] have recently considered more *succinct* descriptions of polynomials where one measures the size of a polynomial by the size of an algebraic circuit needed to compute it. This is potentially much more powerful as there are polynomials such as the determinant which are of high degree and involve exponentially many monomials and yet can be computed by small algebraic circuits. They named the resulting system the *Ideal Proof System (IPS)* which we now define.

▶ **Definition 2.1** (Ideal Proof System (IPS), Grochow-Pitassi [26]). Let $f_1(\vec{x}), \ldots, f_m(\vec{x}) \in \mathbb{F}[x_1, \ldots, x_n]$ be a system of polynomials. An **IPS refutation** for showing that the polynomials $\{f_j\}_j$ have no common solution in $\{0,1\}^n$ is an algebraic circuit $C(\vec{x}, \vec{y}, \vec{z}) \in \mathbb{F}[\vec{x}, y_1, \ldots, y_m, z_1, \ldots, z_n]$, such that

1. $C(\vec{x}, \vec{0}, \vec{0}) = 0$.
2. $C(\vec{x}, f_1(\vec{x}), \ldots, f_m(\vec{x}), x_1^2 - x_1, \ldots, x_n^2 - x_n) = 1$.

The **size** of the IPS refutation is the size of the circuit C. If C is of individual degree ≤ 1 in each y_j and z_i, then this is a **linear** IPS refutation (called *Hilbert* IPS by Grochow-Pitassi [26]), which we will abbreviate as $\mathrm{IPS}_{\mathrm{LIN}}$. If C is of individual degree ≤ 1 only in the y_j then we say this is a $\mathrm{IPS}_{\mathrm{LIN}'}$ refutation. If C comes from a restricted class of algebraic circuits $\mathcal{C}$, then this is a called a $\mathcal{C}$-IPS refutation, and further called a $\mathcal{C}$-$\mathrm{IPS}_{\mathrm{LIN}}$ refutation if C is linear in $\vec{y}, \vec{z}$, and a $\mathcal{C}$-$\mathrm{IPS}_{\mathrm{LIN}'}$ refutation if C is linear in $\vec{y}$.

Notice also that our definition here adds the equations $\{x_i^2 - x_i\}_i$ to the system $\{f_j\}_j$. For convenience we will often denote the equations $\{x_i^2 - x_i\}_i$ as $\vec{x}^2 - \vec{x}$. One need not add the equations $\vec{x}^2 - \vec{x}$ to the system in general, but this is the most interesting regime for proof complexity and thus we adopt it as part of our definition.

though it is a complete refutation system for the standard polynomial translation of unsatisfiable CNFs) but that the $\mathrm{IPS}_{\mathrm{LIN}'}$ version is complete.

Grochow-Pitassi [26] proved the following theorem, showing that the IPS system has surprising power and that lower bounds on this system give rise to *computational* lower bounds.

▶ **Theorem 2.2** (Grochow-Pitassi [26]). *Let φ be a 3CNF. If there is an Extended Frege proof (Frege proof) that φ is unsatisfiable in size-s, then there is an IPS refutation of circuit (formula) size* $\mathsf{poly}(|\varphi|, s)$ *that is checkable in randomized* $\mathsf{poly}(|\varphi|, s)$ *time. Conversely, if every IPS refutation requires circuit (formula) size $\geq s$ then there is an explicit polynomial (that is, in* VNP*) that requires $\geq s$-size algebraic circuits (formulas).*[1]

▶ Remark. One point to note is that the transformation from Extended Frege to IPS refutations yields circuits of polynomial size but without any guarantee on their degree. In particular, such circuits can compute polynomials of exponential degree. In contrast, the conversion from Frege to IPS refutations yields polynomial sized algebraic formulas and those compute polynomials of polynomially bounded degree. This range of parameters, polynomials of polynomially bounded degree, is the more common setting studied in algebraic complexity.

The fact that $\mathcal{C}$-IPS refutations are efficiently checkable (with randomness) follows from the fact that we need to verify the polynomial identities stipulated by the definition. That is, one needs to solve an instance of the *polynomial identity testing (PIT)* problem for the class $\mathcal{C}$: given a circuit from the class $\mathcal{C}$ decide whether it computes the identically zero polynomial.

[1] We note that Grochow and Pitassi [26] proved this for Extended Frege and circuits, but essentially the same proof relates Frege and formula size.

This problem is solvable in probabilistic polynomial time (BPP) for general algebraic circuits, and there are various restricted classes for which deterministic algorithms are known.

Motivated by the fact that PIT of non-commutative formulas[2] can be solved deterministically ([47]) and admit exponential-size lower bounds ([38]), Li, Tzameret and Wang [37] have shown that IPS over *non-commutative* polynomials can simulate Frege (they also provided a quasipolynomial simulation of IPS over non-commutative formulas by Frege; see Li, Tzameret and Wang [37] for more details).

▶ **Theorem 2.3** (Li, Tzameret and Wang [37]). *Let φ be a 3CNF. If Frege can prove that φ is unsatisfiable in size-s, then there is a non-commutative IPS refutation of formula size* $\mathsf{poly}(|\varphi|, s)$ *computing a polynomial of degree* $\mathsf{poly}(|\varphi|, s)$*, where the commutator axioms $x_i x_j - x_j x_i$ are also included in the polynomial system being refuted. Further, this refutation is checkable in deterministic* $\mathsf{poly}(|\varphi|, s)$ *time.*

The above results naturally motivate studying $\mathcal{C}$-IPS for various restricted classes of algebraic circuits, as lower bounds for such proofs then intuitively correspond to restricted lower bounds for the Extended Frege proof system. In particular, as exponential lower bounds are known for non-commutative formulas ([38]), this possibly suggests that such methods could even attack the full Frege system itself.

3 Algebraic Circuit Classes

Having motivated $\mathcal{C}$-IPS for restricted circuit classes $\mathcal{C}$, we now give formal definitions of the algebraic circuit classes of interest to this paper, all of which were studied independently in algebraic complexity. Some of them define the state-of-art in our ability to prove lower bounds and provide efficient deterministic identity tests, so it is natural to attempt converting these to the proof complexity framework. We define each and briefly explain what we know about it. As the list is long, the reader may consider skipping to the results (Section 4), and refer to the definitions of these classes as they arise.

Algebraic circuits and formula (over a fixed chosen field) compute polynomials via addition and multiplication gates, starting from the input variables and constants from the field. For background on algebraic circuits in general and their complexity measures we refer the reader to the survey [54]. We next define the restricted circuit classes that we will be studying in this paper.

3.1 Low Depth Classes

We start by defining what are the simplest and most restricted classes of algebraic circuits. The first class simply represents polynomials as a sum of monomials. This is also called the *sparse representation* of the polynomial. Notationally we call this model $\sum\prod$ formulas (to capture the fact that polynomials computed in the class are represented simply as sums of products), but we will more often call these polynomials "sparse".

▶ **Definition 3.1.** The class $\mathcal{C} = \sum\prod$ compute polynomials in their sparse representation, i.e., as sum of monomials. The graph of computation has two layers with an addition gate at the top and multiplication gates at the bottom. The size of a $\sum\prod$ circuit of a polynomial f is the number of monomials in f.

[2] These are formulas over a set of non-commuting variables.

This class of circuits is what is used in the Nullstellensatz proof system. In our terminology $\sum\prod$-IPS$_{\text{LIN}}$ is exactly the Nullstellensatz proof system.

Another restricted class of algebraic circuits is that of *depth-3 powering formulas* (sometimes also called "diagonal depth-3 circuits"). We will sometimes abbreviate this name as a "$\sum\bigwedge\sum$ formula", where $\bigwedge$ denotes the powering operation. Specifically, polynomials that are efficiently computed by small formulas from this class can be represented as sum of powers of linear functions. This model appears implicitly in Shpilka [53] and explicitly in the work of Saxena [52].

▶ **Definition 3.2.** The class of depth-3 powering formulas, denoted $\sum\bigwedge\sum$, computes polynomials of the following form

$$f(\vec{x}) = \sum_{i=1}^{s} \ell_i(\vec{x})^{d_i},$$

where $\ell_i(\vec{x})$ are linear functions. The degree of this $\sum\bigwedge\sum$ representation of f is $\max_i\{d_i\}$ and its size is $n \cdot \sum_{i=1}^{s}(d_i+1)$.

One reason for considering this class of circuits is that it is a simple, but non-trivial model that is somewhat well-understood. In particular, the partial derivative method of Nisan-Wigderson [40] implies lower bounds for this model and efficient PIT algorithms are known ([52, 3, 21, 22, 19]).

We also consider a generalization of this model where we allow powering of low-degree polynomials.

▶ **Definition 3.3.** The class $\sum\bigwedge\sum\prod^{t}$ computes polynomials of the following form

$$f(\vec{x}) = \sum_{i=1}^{s} f_i(\vec{x})^{d_i} \ ,$$

where the degree of the $f_i(\vec{x})$ is at most t. The size of this representation is $\binom{n+t}{t} \cdot \sum_{i=1}^{s}(d_i+1)$.

We remark that the reason for defining the size this way is that we think of the f_i as represented as sum of monomials (there are $\binom{n+t}{t}$ n-variate monomials of degree at most t) and the size captures the complexity of writing this as an algebraic formula. This model is the simplest that requires the method of *shifted partial derivatives* of Kayal [34, 27] to establish lower bounds, and this has recently been generalized to obtain PIT algorithms ([16]).

3.2 Oblivious Algebraic Branching Programs

Algebraic branching programs (ABPs) form a model whose computational power lies between that of algebraic circuits and algebraic formulas, and certain *read-once* and *oblivious* ABPs are a natural setting for studying the *partial derivative matrix* lower bound technique of Nisan [38].

▶ **Definition 3.4** (Nisan [38])**.** An **algebraic branching program (ABP) with unrestricted weights** of depth D and width $\leq r$, on the variables $x_1, \ldots, x_n$, is a directed acyclic graph such that:

- The vertices are partitioned in $D+1$ layers $V_0, \ldots, V_D$, so that $V_0 = \{s\}$ (s is the source node), and $V_D = \{t\}$ (t is the sink node). Further, each edge goes from V_{i-1} to V_i for some $0 < i \leq D$.

- $\max |V_i| \leq r$.
- Each edge e is weighted with a polynomial $f_e \in \mathbb{F}[\vec{x}]$.

Each s-t path is said to compute the polynomial which is the product of the labels of its edges, and the algebraic branching program itself computes the sum over all s-t paths of such polynomials.

- An algebraic branching program is said to be **oblivious** if for every layer ℓ, all the edge labels in that layer are univariate polynomials in a variable x_{i_ℓ}.
- An oblivious branching program is said to be a **read-once** oblivious ABP (roABP) if the x_{i_ℓ}'s are distinct variables, so that $D = n$. That is, each x_i appears in the edge labels in at exactly one layer. The layers thus define a **variable order**, which will be $x_1 < \cdots < x_n$ if not otherwise specified.
- An oblivious branching program is said to be a **read-k** oblivious ABP if each variable x_i appears in the edge labels of at most k layers, so that $D = kn$.
- An ABP is non-commutative if it is defined over the ring of non-commuting variables.

Intuitively, roABPs are the algebraic analog of read-once boolean branching program, the non-uniform model of the class RL. Nisan [38] proved lower bounds for non-commutative ABPs (and thus also for roABPs, in any order) and in a sequence of papers polynomial identity testing algorithms were devised for it ([47, 22, 19, 2]). Recently Anderson, Forbes, Saptharishi, Shpilka, and Volk [5] obtained exponential lower bounds for read-k oblivious ABPs (when $k = o(\log n / \log \log n)$) as well as a slightly subexponential PIT algorithm.

We note that roABPs are known to simulate non-commutative formulas ([38]). Thus, the result of Li, Tzameret and Wang [37] (see Theorem 2.3) demonstrates the importance of studying IPS proofs over roABPs (see also Tzameret [56]).

3.3 Multilinear Formulas

The last model that we consider is that of multilinear formulas.

▶ **Definition 3.5** (Multilinear formula)**.** An algebraic formula is *a multilinear formula* (or equivalently, *multilinear algebraic formula*) if the polynomial computed by *each* gate of the formula is multilinear (as a formal polynomial, that is, as an element of $\mathbb{F}[x_1, \ldots, x_n]$).

Raz [46, 45] proved quasi-polynomial lower bounds for multilinear formulas and separated multilinear formulas from multilinear circuits. Raz and Yehudayoff proved exponential lower bounds for small depth multilinear formulas [49]. Only slightly sub-exponential polynomial identity testing algorithms are known for small-depth multilinear formulas ([42]).

4 Our Results and Techniques

We now briefly summarize our results and techniques, stating some results in less than full generality to more clearly convey the result. We present the results in the order that we later prove them. We start by giving upper bounds for the IPS (Subsection 4.1). We then describe our functional lower bounds and the IPS_{LIN} lower bounds they imply (Subsection 4.2). Finally, we discuss lower bounds for multiples and state our lower bounds for IPS (Subsection 4.3).

4.1 Upper Bounds for Proofs within Subclasses of IPS

Grochow and Pitassi [26] showed that the full IPS proof system can simulate powerful proof systems such as Extended Frege. This left open the extent to which $\mathcal{C}$-IPS can refute

interesting sets of polynomial equations for restricted classes $\mathcal{C}$. We establish here that even restricted classes of IPS are powerful, such as being able to refute interesting unsatisfiable systems of equations arising from particular instances of NP-complete problems.

Our first upper bound is to show that linear-IPS can simulate the full IPS proof system when the axioms are computationally simple.

▶ **Theorem 4.1.** *For $|\mathbb{F}| \geq \mathsf{poly}(d)$, if $f_1, \ldots, f_m \in \mathbb{F}[x_1, \ldots, x_n]$ are degree-d polynomials computable by size-s algebraic circuits and they have a size-t IPS refutation, then they also have a size-$\mathsf{poly}(d, s, t)$ IPS$_{LIN}$ refutation.*

This theorem is established by pushing the "non-linear" dependencies on the axioms into the IPS refutation itself, which is possible as the axioms are assumed to themselves be computable by small circuits. We note that Grochow and Pitassi [26] showed such a conversion, but only for IPS refutations computable by sparse polynomials.

We then turn our attention to IPS involving only restricted classes of algebraic circuits, and show that they are complete proof systems. This is clear for complete models of algebraic circuits such as sparse polynomials, depth-3 powering formulas [3] and roABPs. For multilinear formulas this is more subtle as not every polynomial is multilinear, however we can show a simulation of sparse-IPS$_{\text{LIN}}$ by a careful multilinearization.

▶ **Theorem 4.2.** *The proof systems of sparse-IPS$_{LIN}$, $\sum\bigwedge\sum$-IPS$_{LIN}$ (in large characteristic fields), and roABP-IPS$_{LIN}$ are complete proof systems (for systems of polynomials with no boolean solutions). The multilinear-formula-IPS$_{LIN}$ proof system is not complete, but the depth-2 multilinear-formula-IPS$_{LIN'}$ proof system is complete (for multilinear axioms) and can polynomially simulate sparse-IPS$_{LIN}$ (for low-degree axioms). For standard polynomial* translation of CNFs*, multilinear-formula-IPS$_{LIN}$ is complete (even without using the boolean axioms).*

We next consider the equation $\sum_{i=1}^{n} \alpha_i x_i - \beta$ along with the boolean axioms $\{x_i^2 - x_i\}_i$. Deciding whether this system of equations is satisfiable is the NP-complete *subset-sum* problem, and as such we do not expect small refutations in general (unless NP = coNP). Indeed, Impagliazzo, Pudlák, and Sgall [30] have shown lower bounds for refutations in the *polynomial calculus* system (and thus also the Nullstellensatz system) even when $\vec{\alpha} = \vec{1}$. Specifically, they showed that such refutations require both $\Omega(n)$-degree and $\exp(\Omega(n))$-many monomials. In the language of this paper, they gave $\exp(\Omega(n))$-size lower bounds for refuting this system in $\sum\prod$-IPS$_{\text{LIN}}$ (which is equivalent to the Nullstellensatz proof system). In contrast, we establish here $\mathsf{poly}(n)$-size refutations for $\vec{\alpha} = \vec{1}$ in the restricted proof systems of roABP-IPS$_{\text{LIN}}$ and *depth-3* multilinear-formula-IPS$_{\text{LIN}}$ (despite the fact that multilinear-formula-IPS$_{\text{LIN}}$ is not complete).

▶ **Theorem 4.3.** *Let $\mathbb{F}$ be a field of characteristic* $\text{char}(\mathbb{F}) > n$*. Then the system of polynomial equations $\sum_{i=1}^{n} x_i - \beta$, $\{x_i^2 - x_i\}_{i=1}^{n}$ is unsatisfiable for $\beta \in \mathbb{F} \setminus \{0, \ldots, n\}$, and there are explicit $\mathsf{poly}(n)$-size refutations within roABP-IPS$_{LIN}$, as well as within depth-3 multilinear-formula-IPS$_{LIN}$.*

This theorem is proven by noting that the polynomial $p(t) := \prod_{k=0}^{n}(t-k)$ vanishes on $\sum_i x_i$ modulo $\{x_i^2 - x_i\}_{i=1}^{n}$, but $p(\beta)$ is a non-zero constant. This implies that $\sum_i x_i - \beta$ divides $p(\sum_i x_i) - p(\beta)$. Denoting the quotient by $f(\vec{x})$, it follows that $\frac{1}{-p(\beta)} \cdot f(\vec{x}) \cdot (\sum_i x_i - \beta) \equiv 1$

[3] Showing that depth-3 powering formulas are complete (in large characteristic) can be seen from the fact that any multilinear monomial can be computed in this model, see for example Fischer [15].

mod $\{x_i^2 - x_i\}_{i=1}^n$, which is nearly a linear-IPS refutation except for the complexity of establishing this relation over the boolean cube. We show that the quotient f is easily expressed as a depth-3 powering circuit. Unfortunately, proving the above equivalence to 1 modulo the boolean cube is not possible in the depth-3 powering circuit model. However, by moving to more powerful models (such as roABPs and multilinear formulas) we can give proofs of this multilinearization to 1 and thus give proper IPS refutations.

4.2 Linear-IPS Lower Bounds via Functional Lower Bounds

Having demonstrated the power of various restricted classes of IPS refutations by refuting the subset-sum axiom, we now turn to lower bounds. We give two paradigms for establishing lower bounds, the first of which we discus here, named a *functional circuit lower bound*. This term appeared in the work of Grigoriev and Razborov [25] as well as in the recent work of Forbes, Kumar and Saptharishi [18]. We briefly motivate this type of lower bound as a topic of independent interest in algebraic circuit complexity, and then discuss the lower bounds we obtain and their implications to obtaining proof complexity lower bounds.

In algebraic complexity one computes polynomials *syntactically* as objects in the ring $\mathbb{F}[x_1, \ldots, x_n]$. Thus, even if one is only interested in evaluating the polynomial over the boolean cube, yielding a function $\hat{f} : \{0,1\}^n \to \mathbb{F}$, an algebraic computation of the polynomial necessarily gives a method for evaluating the polynomial over $\mathbb{F}$ as well as any extension of $\mathbb{F}$. However, some polynomials such as the permanent are known in boolean complexity to have complex behavior as functions even over boolean inputs, so one would expect that *any* polynomial f that agrees with the permanent on boolean inputs must require large algebraic circuits. We call such results functional circuit lower bounds. Prior work ([24, 25, 36]) has established functional lower bounds over fixed-size finite fields, and the recent work of Forbes, Kumar and Saptharishi [18] has established some lower bounds for any field.

▶ **Goal 4.4** (Functional Circuit Lower Bound ([25, 18])). *Obtain explicit functions $\hat{f} : \{0,1\}^n \to \mathbb{F}$ such that for any polynomial $f \in \mathbb{F}[x_1, \ldots, x_n]$ such that $f(\vec{x}) = \hat{f}(\vec{x})$ for all $\vec{x} \in \{0,1\}^n$, it must be that f requires large algebraic circuits.*

While it is natural to hope that existing methods would yield such lower bounds, many lower bound techniques inherently use that algebraic computation is *syntactic*. In particular, techniques involving partial derivatives (which include the partial derivative method of Nisan-Wigderson [40] and the shifted partial derivative method of Kayal [34, 27]) cannot as is yield functional lower bounds as knowing a polynomial on $\{0,1\}^n$ is not enough to conclude information about its partial derivatives.

We now explain how functional lower bounds imply lower bounds for linear-IPS refutations in certain cases. Suppose one considers refutations of the unsatisfiable polynomial system $f(\vec{x}), \{x_i^2 - x_i\}_{i=1}^n$. A linear-IPS refutation would yield an equation of the form $g(\vec{x}) \cdot f(\vec{x}) + \sum_i h_i(\vec{x}) \cdot (x_i^2 - x_i) = 1$ for some polynomials $g, h_i \in \mathbb{F}[\vec{x}]$. Viewing this equation modulo the boolean cube, we have that $g(\vec{x}) \cdot f(\vec{x}) \equiv 1 \mod \{x_i^2 - x_i\}_i$. Equivalently, since $f(\vec{x})$ is unsatisfiable over $\{0,1\}^n$, we see that $g(\vec{x}) = 1/f(\vec{x})$ for $\vec{x} \in \{0,1\}^n$, as $f(\vec{x})$ is never zero so this fraction is well-defined. It follows that *if* the function $\vec{x} \mapsto 1/f(\vec{x})$ induces a functional lower bound then $g(\vec{x})$ must require large complexity, yielding the desired linear-IPS lower bound.

Thus, it remains to instantiate this program. While we are successful, we should note that this approach as is seems to only yield proof complexity lower bounds for systems with one non-boolean axiom and thus cannot encode polynomial systems arising from 3CNFs in a meaningful way.

Our starting point is to observe that the subset-sum axiom already induces a weak form of functional lower bound, where the complexity is measured by degree.

▶ **Theorem 4.5.** *Let $\mathbb{F}$ be a field of a characteristic at least* $\mathsf{poly}(n)$ *and $\beta \notin \{0,\dots,n\}$. Then $\sum_i x_i - \beta, \{x_i^2 - x_i\}_i$ is unsatisfiable and any polynomial $f \in \mathbb{F}[x_1,\dots,x_n]$ with $f(\vec{x}) = \frac{1}{\sum_i x_i - \beta}$ for $\vec{x} \in \{0,1\}^n$, satisfies $\deg f \geq n$.*

A lower bound of $\lceil \frac{n}{2} \rceil$ was previously established by Impagliazzo, Pudlák, and Sgall [30], but the bound of 'n' (which is tight) will be crucial for our results.

We then lift this result to obtain lower bounds for stronger models of algebraic complexity. In particular, by replacing "x_i" by "$x_i y_i$" we show that the function $\frac{1}{\sum_i x_i y_i - \beta}$ has maximal *evaluation dimension* between $\vec{x}$ and $\vec{y}$, which is some measure of correlation. This measure is essentially *functional*, so that one can lower bound this measure by understanding the functional behavior of the polynomial on finite sets such as the boolean cube. Our lower bound for evaluation dimension follows by examining the above degree bound. Using known relations between this complexity measure and algebraic circuit classes, we can obtain lower bounds for depth-3 powering linear-IPS.

▶ **Theorem 4.6.** *Let $\mathbb{F}$ be a field of characteristic $\geq$* $\mathsf{poly}(n)$ *and $\beta \notin \{0,\dots,n\}$. Then $\sum_{i=1}^n x_i y_i - \beta, \{x_i^2 - x_i\}_i, \{y_i^2 - y_i\}_i$ is unsatisfiable and any $\sum\bigwedge\sum$-IPS$_{LIN}$ refutation requires size $\geq \exp(\Omega(n))$.*

The above axiom only gets maximal correlation between a *fixed* partition of the variables. By introducing auxiliary variables we can create such correlation between *any* partition of (some) of the variables. By again invoking results showing such structure implies computational hardness we obtain more linear-IPS lower bounds.

▶ **Theorem 4.7.** *Let $\mathbb{F}$ be a field of characteristic $\geq$* $\mathsf{poly}(n)$ *and $\beta \notin \{0,\dots,\binom{2n}{2}\}$. Then $\sum_{i<j} z_{i,j} x_i x_j - \beta, \{x_i^2 - x_i\}_{i=1}^n, \{z_{i,j}^2 - z_{i,j}\}_{i<j}$ is unsatisfiable, and any roABP-IPS$_{LIN}$ refutation (in any variable order) requires $\exp(\Omega(n))$-size. Further, any multilinear-formula-IPS$_{LIN'}$ refutation requires $n^{\Omega(\log n)}$-size, and any depth-$(2d+1)$ multilinear-formula-IPS$_{LIN'}$ refutation requires $n^{\Omega((n/\log n)^{1/d}/d^2)}$-size.*

Thus, we show that even though roABP-IPS$_{\mathrm{LIN}}$ and depth-3 multilinear formula-IPS$_{\mathrm{LIN}'}$ can refute the subset-sum axiom in polynomial size, slight variants of this axiom do not have polynomial-size refutations.

4.3 Lower Bounds for Multiples

While the above paradigm can establish super-polynomial lower bounds for *linear*-IPS, it does not seem able to establish lower bounds for the general IPS proof system, even for restricted classes. This is because such systems would induce equations such as $h(\vec{x})f(\vec{x})^2 + g(\vec{x})f(\vec{x}) \equiv 1 \mod \{x_i^2 - x_i\}_{i=1}^n$, where we need to design a computationally simple axiom f so that this equation implies at least one of h or g is of large complexity. In the linear-IPS case we could assume h was zero, so that we can uniquely solve for $g(\vec{x})$ for $\vec{x} \in \{0,1\}^n$. However, in general knowing $f(\vec{x})$ does not uniquely determine $g(\vec{x})$ or $h(\vec{x})$, which makes this approach significantly more complicated. Further, even though we can efficiently simulate IPS by linear-IPS in general, this simulation increases the complexity of the proof so that even if one started with a $\mathcal{C}$-IPS proof for a restricted circuit class $\mathcal{C}$ the resulting IPS$_{\mathrm{LIN}}$ proof may not be in $\mathcal{C}$-IPS$_{\mathrm{LIN}}$.

As such, we introduce a second paradigm, called *lower bounds for multiples*, which can yield $\mathcal{C}$-IPS lower bounds for various restricted classes $\mathcal{C}$. We begin by defining this question.

▶ **Goal 4.8** (Lower Bounds for Multiples). *Design an explicit polynomial $f(\vec{x})$ such that for any non-zero $g(\vec{x})$ we have that $g(\vec{x})f(\vec{x})$ is hard to compute.*

We now explain how such lower bounds yield IPS lower bounds. Consider the system $f, \{x_i^2 - x_i\}_i$ with a single non-boolean axiom. An IPS refutation is a circuit $C(\vec{x}, y, \vec{z})$ such that $C(\vec{x}, 0, \vec{0}) = 0$ and $C(\vec{x}, f, \vec{x}^2 - \vec{x}) = 1$, where (as mentioned) $\vec{x}^2 - \vec{x}$ denotes $\{x_i^2 - x_i\}_i$. Expressing $C(\vec{x}, f, \vec{x}^2 - \vec{x})$ as a univariate in f, we obtain that $\sum_{i \geq 1} C_i(\vec{x}, \vec{x}^2 - \vec{x}) f^i = 1 - C(\vec{x}, 0, \vec{x}^2 - \vec{x})$ for some polynomials C_i. For many natural measures of circuit complexity $1 - C(\vec{x}, 0, \vec{x}^2 - \vec{x})$ has complexity roughly bounded by that of C itself. Though not strictly necessary for this method, it is worth noting that the complexity of each of the C_i is not much larger than that of C, as one can compute the C_i by homogenizing or interpolating C in the variable y (see for example the survey of Shpilka and Yehudayoff [54]). Thus, we see that a multiple of f has a small circuit, as $\left(\sum_{i \geq 1} C_i(\vec{x}, \vec{x}^2 - \vec{x}) f^{i-1}\right) \cdot f = 1 - C(\vec{x}, 0, \vec{x}^2 - \vec{x})$. Thus, if we can show that all multiples of f require large circuits then we rule out a small IPS refutation.

We now turn to methods for obtaining polynomials with hard multiples. Intuitively if a polynomial f is hard then so should small modifications such as $f^2 + x_1 f$, and this intuition is supported by the result of Kaltofen [32] which shows that if a polynomial has a small algebraic circuit then so do all of its factors. As a consequence, if a polynomial requires super-polynomially large algebraic circuits then so do all of its multiples. However, Kaltofen's [32] result is about *general* algebraic circuits, and there are very limited results about the complexity of factors of *restricted* algebraic circuits ([14, 41]) so that obtaining polynomials for hard multiples via factorization results seems difficult.

However, note that lower bound for multiples has a different order of quantifiers than the factoring question. That is, Kaltofen's [32] result speaks about the factors of *any* small circuit, while the lower bound for multiples speaks about the multiples of a *single* polynomial. Thus, it seems plausible that existing methods could yield such explicit polynomials, and indeed we show this is the case.

We begin by noting that obtaining lower bounds for multiples is a natural instantiation of the algebraic *hardness versus randomness* paradigm. In particular, Heintz-Schnorr [28] and Agrawal [1] showed that obtaining deterministic (black-box) PIT algorithms implies lower bounds, and we strengthen that connection here to lower bounds for multiples. We can actually instantiate this connection, and we use slight modifications of existing PIT algorithms to show that multiples of the determinant are hard in some models.

▶ **Theorem 4.9.** *Let $\mathcal{C}$ be a restricted class of n-variate algebraic circuits. Full derandomization of PIT algorithms for $\mathcal{C}$ yields an explicit polynomials all of whose multiples require $\exp(\Omega(n))$-size as $\mathcal{C}$-circuits.*

In particular, when $\mathcal{C}$ is the class of sparse polynomials, depth-3 powering formulas, $\sum \bigwedge \sum \prod^{\text{‰}(1)}$ formulas (in characteristic zero), or "every-order" roABPs, then all nonzero multiples of the $n \times n$ determinant are $\exp(\Omega(n))$-hard in these models.

The above statement shows that *derandomization* implies *hardness*. We also partly address the converse direction by arguing that hardness-to-randomness construction of Kabanets and Impagliazzo [31] only requires lower bounds for multiples to derandomize PIT. Unfortunately, this direction is harder to instantiate for restricted classes as it requires lower bounds for classes with suitable closure properties.[4]

[4] Although, we note that one can instantiate this connection with depth-3 powering formulas (or even

Unfortunately the above result is slightly unsatisfying from a proof complexity standpoint as the (exponential-size) lower bounds for the subclasses of IPS one can derive from the above result would involve the determinant polynomial as an axiom. While the determinant is efficiently computable, it is not computable by the above restricted circuit classes (indeed, the above result proves that). As such, this would not fit the real goal of proof complexity which seeks to show that there are statements whose proofs must be *super-polynomial larger* than the length of the statement. Thus, if we measure the size of the IPS proof and the axioms with respect to the same circuit measure, the lower bounds for multiples approach *cannot* establish such super-polynomial lower bounds.

However, we believe that lower bounds for multiples could lead, with further ideas, to proof complexity lower bounds in the conventional sense. That is, it seems plausible that by adding *extension variables* we can convert complicated axioms to simple, local axioms by tracing through the computation of that axiom. That is, consider the axiom $xyzw$. This can be equivalently written as $\{a - xy, b - zw, c - ab, c\}$, where this conversion is done by considering a natural algebraic circuit for $xyzw$, replacing each gate with a new variable, and adding an axiom ensuring the new variables respect the computation of the circuit. While we are unable to understand the role of extension variables in this work, we aim to give as simple axioms as possible whose multiples are all hard as this may facilitate future work on extension variables.

We now discuss the lower bounds for multiples we obtain.[5]

▶ **Theorem 4.10.** *We obtain the following lower bounds for multiples.*

- *All non-zero multiples of $x_1 \cdots x_n$ require $\exp(\Omega(n))$-size as a depth-3 powering formula (over any field), or as a $\sum\bigwedge\sum\prod^{\text{‰}(1)}$ formula (in characteristic zero).*
- *All non-zero multiples of $(x_1 + 1) \cdots (x_n + 1)$ require $\exp(\Omega(n))$-many monomials.*
- *All non-zero multiples of $\prod_i (x_i + y_i)$ require $\exp(\Omega(n))$-width as a roABPs in any variable order where $\vec{x}$ precedes $\vec{y}$.*
- *All non-zero multiples of $\prod_{i,j=1}^{n}(z_{i,j} \cdot (x_i + x_j + x_i x_j) + (1 - z_{i,j}))$ require $\exp(\Omega(n))$-width as a roABP in any variable order, as well as $\exp(\Omega(n))$-width as a read-twice oblivious ABP.*

We now briefly explain our techniques for obtaining these lower bounds, focusing on the simplest case of depth-3 powering formulas. It follows from the partial derivative method of Nisan and Wigderson [39] (see Kayal [33]) that such formulas require exponential size to compute the monomial $x_1 \dots x_n$ *exactly*. Forbes and Shpilka [21], in giving a PIT algorithm for this class, showed that this lower bound can be *scaled down* and *made robust*. That is, if one has a size-s depth-3 powering formula, it follows that *if* it computes a monomial $x_{i_1} \cdots x_{i_\ell}$ for distinct i_j then $l \leq O(\log s)$ (so the lower bound is scaled down). One can then show that regardless of what this formula actually computes the *leading* monomial $x_{i_1}^{a_{i_1}} \cdots x_{i_\ell}^{a_{i_\ell}}$ (for distinct i_j and positive a_{i_j}) must have that $\ell \leq O(\log s)$. One then notes that leading monomials are *multiplicative*. Thus, for any non-zero g the leading monomial of $g \cdot x_1 \dots x_n$ involves n variables so that if $g \cdot x_1 \dots x_n$ is computed in size-s then $n \leq O(\log s)$, giving $s \geq \exp(\Omega(n))$ as desired. One can then obtain the other lower bounds using the same

$\sum\bigwedge\sum\prod^{\text{‰}(1)}$ formulas) using the lower bounds for multiples developed in this paper, building on the work of Forbes [17]. However, the resulting PIT algorithms are worse than those developed by Forbes [17].

[5] While we discussed functional lower bounds for multilinear formulas, this class is not interesting for the lower bounds for multiples question. This is because a multiple of a multilinear polynomial may not be multilinear, and thus clearly cannot have a multilinear formula.

idea, though for roABPs one needs to define a leading *diagonal* (refining an argument of Forbes-Shpilka [20]).

We now conclude our IPS lower bounds.

▶ **Theorem 4.11.** *We obtain the following lower bound for subclasses of IPS.*

- *In characteristic zero, for $m \neq n$, the system of polynomials $x_1 \cdots x_n - 1, x_1 + \cdots + x_n - m, \{x_i^2 - x_i\}_{i=1}^n$ is unsatisfiable, any $\sum\bigwedge\sum$-IPS refutation requires $\exp(\Omega(n))$-size.*
- *The system of polynomials, $1 + \prod_{i,j=1}^n (z_{i,j}(x_i + x_j - x_i x_j) + (1 - z_{i,j})), \{x_i^2 - x_i\}_i, \{z_{i,j}^2 - z_{i,j}\}_{i,j}$ is unsatisfiable, and any roABP-IPS refutation (in any variable order) must be of width $\exp(\Omega(n))$.*

Note that the first result is an encoding that $\mathrm{AND}(x_1, \ldots, x_n) = 1$ but the number of variables that equal 1 is different than n. The second is not as natural, but contains the simpler polynomial $\prod_i (u_i + v_i - u_i v_i) + 1$ (up to renaming, and after appropriate substitution of the $z_{i,j}$ to values from $\{0, 1\}$), which encodes that $\mathrm{AND}(\mathrm{OR}(u_1, v_1), \cdots, \mathrm{OR}(u_n, v_n)) \notin \{0, 1\}$.

5 Discussion

In this paper we proved new lower bounds for the Grochow-Pitassi Ideal Proof System (IPS), for various restricted circuit classes underlying this proof system. The main novelty here, as compared with essentially all previous work in algebraic proof complexity, is that lower bounds are proved directly for the most interesting computational complexity measure, namely circuit size, rather than simpler notions of complexity such as degree and sparsity of the polynomials involved. This opens up a path to extending our results to IPS over other circuit classes, in particular ones for which there are already computational lower bounds. A specific challenge is doing so for IPS using depth-4 arithmetic circuits, for which recent exciting work using shifted partial derivatives imply superpolynomial computational lower bounds for natural polynomials.

A different challenge, even for the circuit classes considered here, is that most of our results apply only when the hard contradiction has a specific, and somewhat unnatural structure: aside from the Boolean axioms, there is only one more axiom, which involves all variables (and sometimes also has high degree). Natural tautologies studied in proof complexity arise from k-CNF formulas, where $k = O(1)$, so the contradiction contains many polynomials, each on a constant number of variables (and hence also of constant degree). The techniques in this paper cannot prove IPS lower bounds for such contradictions even with the simplest circuit classes. It would be extremely interesting to devise techniques able to handle such contradictions, arising, e.g., from Tseitin tautologies or random CNFs.

A more specific direction to follow arises from our "PIT technique". In Subsection 4.3 we noted that this technique of using lower bounds for multiples requires including the determinant as an axiom, and it only works for models that cannot efficiently compute the determinant. It would be interesting to weaken this requirement. For example, instead of considering systems that include the determinant as an axiom, one could instead consider the (algebraic) "hard matrix identities" that were suggested by Cook and Rackoff (cf. [7]) and later studied by Soltys and Cook [55]. Recall that the task at hand is, starting from the axioms $XY - I$ (where X and Y are symbolic $n \times n$ matrices), the goal is to derive $YX - I$ in IPS. Here the axioms are easily computable by roABPs, but the derivation is believed to require computing the determinant, so it should be hard for roABP-IPS (see Hrubeš-Tzameret [29] and also Appendix B of the arXiv version of Grochow-Pitassi [26] for more discussion on this.)

Finally, we leave open the question of extending our results from lower bounds on the "static" IPS to lower bounds on a "dynamic" algebraic proof system like the polynomial calculus:

▶ **Open Problem 5.1.** *Can the lower bounds on roABP-IPS$_{LIN}$ and multilinear-formula-IPS$_{LIN}$ from Theorem 4.7 be extended to (tree-like or dag-like) PC over roABPs ([56]) and PC over multilinear formulas (fMC from [48]), respectively?*

Acknowledgments. We would like to thank Rafael Oliviera for helpful discussions, as well as Mrinal Kumar and Ramprasad Saptharishi for conversations [18] clarifying the roles of functional lower bounds in this work. We would also like to thank Joshua Grochow for helpful discussions regarding this work. We are grateful for the CCC'16 reviewers for their careful read of the paper and for their comments.

References

1 Manindra Agrawal. Proving lower bounds via pseudo-random generators. In *Proceedings of the 25th International Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2005)*, pages 92–105, 2005. doi:10.1007/11590156_6.

2 Manindra Agrawal, Rohit Gurjar, Arpita Korwar, and Nitin Saxena. Hitting-sets for ROABP and sum of set-multilinear circuits. *arXiv*, 1406.7535, 2014. URL: http://arxiv.org/abs/1406.7535.

3 Manindra Agrawal, Chandan Saha, and Nitin Saxena. Quasi-polynomial hitting-set for set-depth-Δ formulas. In *Proceedings of the 45th Annual ACM Symposium on Theory of Computing (STOC 2013)*, pages 321–330, 2013. Full version at arXiv:1209.2333. doi:10.1145/2488608.2488649.

4 Michael Alekhnovich and Alexander A. Razborov. Lower bounds for polynomial calculus: Non-binomial case. In *Proceedings of the 42nd Annual IEEE Symposium on Foundations of Computer Science (FOCS 2001)*, pages 190–199, 2001. doi:10.1109/SFCS.2001.959893.

5 Matthew Anderson, Michael A. Forbes, Ramprasad Saptharishi, Amir Shpilka, and Ben Lee Volk. Identity testing and lower bounds for read-k oblivious algebraic branching programs. *Electronic Colloquium on Computational Complexity (ECCC)*, 22:184, 2015. URL: http://eccc.hpi-web.de/report/2015/184/.

6 Paul Beame, Russell Impagliazzo, Jan Krajíček, Toniann Pitassi, and Pavel Pudlák. Lower bounds on Hilbert's Nullstellensatz and propositional proofs. *Proc. London Math. Soc. (3)*, 73(1):1–26, 1996. Preliminary version in the *35th Annual IEEE Symposium on Foundations of Computer Science (FOCS 1994)*. doi:10.1112/plms/s3-73.1.1.

7 Paul Beame and Toniann Pitassi. Propositional proof complexity: past, present, and future. *Bull. Eur. Assoc. Theor. Comput. Sci. EATCS*, 1998(65):66–89, 1998.

8 Samuel R. Buss, Dima Grigoriev, Russell Impagliazzo, and Toniann Pitassi. Linear gaps between degrees for the polynomial calculus modulo distinct primes. *J. Comput. Syst. Sci.*, 62(2):267–289, 2001. Preliminary version in the *14th Annual IEEE Conference on Computational Complexity (CCC 1999)*. doi:10.1006/jcss.2000.1726.

9 Samuel R. Buss, Russell Impagliazzo, Jan Krajíček, Pavel Pudlák, Alexander A. Razborov, and Jiří Sgall. Proof complexity in algebraic systems and bounded depth Frege systems with modular counting. *Computational Complexity*, 6(3):256–298, 1996. doi:10.1007/BF01294258.

10 Matthew Clegg, Jeff 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 1996)*, pages 174–183, 1996. doi:10.1145/237814.237860.

11 Stephen A. Cook and Robert A. Reckhow. On the lengths of proofs in the propositional calculus (preliminary version). In *Proceedings of the 6th Annual ACM Symposium on Theory of Computing (STOC 1974)*, pages 135–148, 1974. For corrections see Cook-Reckhow [12]. doi:10.1145/800119.803893.

12 Stephen A. Cook and Robert A. Reckhow. Corrections for "On the lengths of proofs in the propositional calculus (preliminary version)". *SIGACT News*, 6(3):15–22, July 1974. doi:10.1145/1008311.1008313.

13 Stephen A. Cook and Robert A. Reckhow. The relative efficiency of propositional proof systems. *J. Symb. Log.*, 44(1):36–50, 1979. This is a journal-version of Cook-Reckhow [11] and Reckhow [51]. doi:10.2307/2273702.

14 Zeev Dvir, Amir Shpilka, and Amir Yehudayoff. Hardness-randomness tradeoffs for bounded depth arithmetic circuits. *SIAM J. Comput.*, 39(4):1279–1293, 2009. Preliminary version in the *40th Annual ACM Symposium on Theory of Computing (STOC 2008)*. doi:10.1137/080735850.

15 Ismor Fischer. Sums of like powers of multivariate linear forms. *Mathematics Magazine*, 67(1):59–61, 1994. URL: http://www.jstor.org/stable/2690560.

16 Michael A. Forbes. *Polynomial Identity Testing of Read-Once Oblivious Algebraic Branching Programs*. PhD thesis, Massachusetts Institute of Technology, June 2014. URL: http://hdl.handle.net/1721.1/89843.

17 Michael A. Forbes. Deterministic divisibility testing via shifted partial derivatives. In *Proceedings of the 56th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2015)*, 2015.

18 Michael A. Forbes, Mrinal Kumar, and Ramprasad Saptharishi. Functional lower bounds for arithmetic circuits and boolean circuit complexity. Manuscript, 2015.

19 Michael A. Forbes, Ramprasad Saptharishi, and Amir Shpilka. Hitting sets for multilinear read-once algebraic branching programs, in any order. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC 2014)*, pages 867–875, 2014. Full version at arXiv:1309.5668. doi:10.1145/2591796.2591816.

20 Michael A. Forbes and Amir Shpilka. On identity testing of tensors, low-rank recovery and compressed sensing. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC 2012)*, pages 163–172, 2012. Full version at arXiv:1111.0663. doi:10.1145/2213977.2213995.

21 Michael A. Forbes and Amir Shpilka. Explicit Noether Normalization for simultaneous conjugation via polynomial identity testing. In *Proceedings of the 17th International Workshop on Randomization and Computation (RANDOM 2013)*, pages 527–542, 2013. Full version at arXiv:1303.0084. doi:10.1007/978-3-642-40328-6_37.

22 Michael A. Forbes and Amir Shpilka. Quasipolynomial-time identity testing of noncommutative and read-once oblivious algebraic branching programs. In *Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2013)*, pages 243–252, 2013. Full version at arXiv:1209.2408. doi:10.1109/FOCS.2013.34.

23 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 1998)*, pages 648–652, 1998. doi:10.1109/SFCS.1998.743515.

24 Dima Grigoriev and Marek Karpinski. An exponential lower bound for depth 3 arithmetic circuits. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing (STOC 1998)*, pages 577–582, 1998. doi:10.1145/276698.276872.

25 Dima Grigoriev and Alexander A. Razborov. Exponential lower bounds for depth 3 arithmetic circuits in algebras of functions over finite fields. *Appl. Algebra Eng. Commun. Comput.*, 10(6):465–487, 2000. Preliminary version in the *39th Annual IEEE Symposium on Foundations of Computer Science (FOCS 1998)*. doi:10.1007/s002009900021.

26 Joshua A. Grochow and Toniann Pitassi. Circuit complexity, proof complexity, and polynomial identity testing. In *Proceedings of the 55th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2014)*, pages 110–119, 2014. Full version at arXiv:abs/1404.3820. doi:10.1109/FOCS.2014.20.

27 Ankit Gupta, Pritish Kamath, Neeraj Kayal, and Ramprasad Saptharishi. Approaching the chasm at depth four. *J. ACM*, 61(6):33:1–33:16, December 2014. Preliminary version in the *28th Annual IEEE Conference on Computational Complexity (CCC 2013)*. doi:10.1145/2629541.

28 Joos Heintz and Claus-Peter Schnorr. Testing polynomials which are easy to compute (extended abstract). In *Proceedings of the 12th Annual ACM Symposium on Theory of Computing (STOC 1980)*, pages 262–272, 1980. doi:10.1145/800141.804674.

29 Pavel Hrubes and Iddo Tzameret. Short proofs for the determinant identities. *SIAM J. Comput.*, 44(2):340–383, 2015. Preliminary version in the *44th Annual ACM Symposium on Theory of Computing (STOC 2012)*. doi:10.1137/130917788.

30 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. doi:10.1007/s000370050024.

31 Valentine Kabanets and Russell Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Computational Complexity*, 13(1-2):1–46, 2004. Preliminary version in the *35th Annual ACM Symposium on Theory of Computing (STOC 2003)*. doi:10.1007/s00037-004-0182-6.

32 Erich L. Kaltofen. Factorization of polynomials given by straight-line programs. In Silvio Micali, editor, *Randomness and Computation*, volume 5 of *Advances in Computing Research*, pages 375–412. JAI Press, Inc., Greenwich, CT, USA, 1989. URL: http://www.math.ncsu.edu/~kaltofen/bibliography/89/Ka89_slpfac.pdf.

33 Neeraj Kayal. Personal Communication to Saxena [52], 2008.

34 Neeraj Kayal. An exponential lower bound for the sum of powers of bounded degree polynomials. *Electronic Colloquium on Computational Complexity (ECCC)*, 19(81), 2012. URL: http://eccc.hpi-web.de/report/2012/081.

35 Jan Krajíček. *Bounded arithmetic, propositional logic, and complexity theory*, volume 60 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, Cambridge, 1995. doi:10.1017/CBO9780511529948.

36 Mrinal Kumar and Ramprasad Saptharishi. An exponential lower bound for homogeneous depth-5 circuits over finite fields. *arXiv*, 1507.00177, 2015. URL: http://arxiv.org/abs/1507.00177.

37 Fu Li, Iddo Tzameret, and Zhengyu Wang. Non-commutative formulas and Frege lower bounds: a new characterization of propositional proofs. In *Proceedings of the 30th Computational Complexity Conference (CCC), June 17-19, 2015*, 2015.

38 Noam Nisan. Lower bounds for non-commutative computation. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing (STOC 1991)*, pages 410–418, 1991. doi:10.1145/103418.103462.

39 Noam Nisan and Avi Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994. Preliminary version in the *29th Annual IEEE Symposium on Foundations of Computer Science (FOCS 1988)*. doi:10.1016/S0022-0000(05)80043-1.

40 Noam Nisan and Avi Wigderson. Lower bounds on arithmetic circuits via partial derivatives. *Computational Complexity*, 6(3):217–234, 1996. Preliminary version in the *36th Annual IEEE Symposium on Foundations of Computer Science (FOCS 1995)*. doi:10.1007/BF01294256.

41 Rafael Oliveira. Factors of low individual degree polynomials. In *Proceedings of the 30th Annual Computational Complexity Conference (CCC 2015)*, volume 33 of *Leibniz Inter-*

national Proceedings in Informatics (LIPIcs), pages 198–216, 2015. doi:10.4230/LIPIcs.CCC.2015.198.

42 Rafael Oliveira, Amir Shpilka, and Ben Lee Volk. Subexponential size hitting sets for bounded depth multilinear formulas. In *Proceedings of the 30th Annual Computational Complexity Conference (CCC 2015)*, volume 33 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 304–322, 2015. Full version at arXiv:1411.7492. doi:10.4230/LIPIcs.CCC.2015.304.

43 Toniann Pitassi. Algebraic propositional proof systems. In *Descriptive complexity and finite models (Princeton, NJ, 1996)*, volume 31 of *DIMACS Ser. Discrete Math. Theoret. Comput. Sci.*, pages 215–244. Amer. Math. Soc., Providence, RI, 1997.

44 Pavel Pudlák. Lower bounds for resolution and cutting plane proofs and monotone computations. *The Journal of Symbolic Logic*, 62(3):981–998, Sept. 1997.

45 Ran Raz. Separation of multilinear circuit and formula size. *Theory of Computing*, 2(6):121–135, 2006. Preliminary version in the *45th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2004)*. doi:10.4086/toc.2006.v002a006.

46 Ran Raz. Multi-linear formulas for permanent and determinant are of super-polynomial size. *J. ACM*, 56(2), 2009. Preliminary version in the *36th Annual ACM Symposium on Theory of Computing (STOC 2004)*. doi:10.1145/1502793.1502797.

47 Ran Raz and Amir Shpilka. Deterministic polynomial identity testing in non-commutative models. *Comput. Complex.*, 14(1):1–19, April 2005. Preliminary version in the *19th Annual IEEE Conference on Computational Complexity (CCC 2004)*. doi:10.1007/s00037-005-0188-8.

48 Ran Raz and Iddo Tzameret. The strength of multilinear proofs. *Computational Complexity*, 17(3):407–457, 2008.

49 Ran Raz and Amir Yehudayoff. Lower bounds and separations for constant depth multilinear circuits. *Computational Complexity*, 18(2):171–207, 2009. Preliminary version in the *23rd Annual IEEE Conference on Computational Complexity (CCC 2008)*. doi:10.1007/s00037-009-0270-8.

50 Alexander A. Razborov. Lower bounds for the polynomial calculus. *Computational Complexity*, 7(4):291–324, 1998. doi:10.1007/s000370050013.

51 Robert A. Reckhow. *On the lengths of proofs in the propositional calculus*. PhD thesis, University of Toronto, 1976. URL: https://www.cs.toronto.edu/~sacook/homepage/reckhow_thesis.pdf.

52 Nitin Saxena. Diagonal circuit identity testing and lower bounds. In *Proceedings of the 35th International Colloquium on Automata, Languages and Programming (ICALP 2008)*, pages 60–71, 2008. Preliminary version in the Electronic Colloquium on Computational Complexity (ECCC), Technical Report TR07-124. doi:10.1007/978-3-540-70575-8_6.

53 Amir Shpilka. Affine projections of symmetric polynomials. *J. Comput. Syst. Sci.*, 65(4):639–659, 2002. Preliminary version in the *16th Annual IEEE Conference on Computational Complexity (CCC 2001)*. doi:10.1016/S0022-0000(02)00021-1.

54 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. doi:10.1561/0400000039.

55 Michael Soltys and Stephen Cook. The proof complexity of linear algebra. *Ann. Pure Appl. Logic*, 130(1-3):277–323, 2004.

56 Iddo Tzameret. Algebraic proofs over noncommutative formulas. *Information and Computation*, 209(10):1269–1292, 2011.

Functional Lower Bounds for Arithmetic Circuits and Connections to Boolean Circuit Complexity

Michael A. Forbes[*1], **Mrinal Kumar**[†2], **and Ramprasad Saptharishi**[‡3]

1 **Department of Computer Science, Princeton University, USA**
miforbes@csail.mit.edu
2 **Department of Computer Science, Rutgers University, USA**
mrinal.kumar@rutgers.edu
3 **Tel Aviv University, Israel**
ramprasad@cmi.ac.in

Abstract

We say that a circuit C over a field $\mathbb{F}$ *functionally* computes a polynomial $P \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ if for every $\mathbf{x} \in \{0,1\}^n$ we have that $C(x) = P(x)$. This is in contrast to *syntactically* computing P, when $C \equiv P$ as formal polynomials. In this paper, we study the question of proving lower bounds for homogeneous depth-3 and depth-4 arithmetic circuits for functional computation. We prove the following results :

- Exponential lower bounds for homogeneous depth-3 arithmetic circuits for a polynomial in VNP.
- Exponential lower bounds for homogeneous depth-4 arithmetic circuits with bounded individual degree for a polynomial in VNP.

Our main motivation for this line of research comes from our observation that strong enough functional lower bounds for even very special depth-4 arithmetic circuits for the Permanent imply a separation between #P and ACC^0. Thus, improving the second result to get rid of the *bounded individual degree* condition could lead to substantial progress in boolean circuit complexity. Besides, it is known from a recent result of Kumar and Saptharishi [9] that over constant sized finite fields, strong enough *average case* functional lower bounds for homogeneous depth-4 circuits imply superpolynomial lower bounds for homogeneous depth-5 circuits.

Our proofs are based on a family of new complexity measures called *shifted evaluation dimension*, and might be of independent interest.

1998 ACM Subject Classification F.2.1 Numerical Algorithms and Problems, I.1.1 Expressions and Their Representation

Keywords and phrases boolean circuits, arithmetic circuits, lower bounds, functional computation

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.33

1 Introduction

Arithmetic circuits are one of the most natural models of computation for studying computation with multivariate polynomials. One of the most fundamental questions in this area

* Research supported by the Princeton Center for Theoretical Computer Science.
† Research supported in part by the Simons Graduate Fellowship.
‡ 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.

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 33; pp. 33:1–33:19

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

of research is to show that there are low degree polynomials which cannot be efficiently computed by *small sized* arithmetic circuits. However, in spite of the significance of this question, progress on it has been sparse and our current state of understanding of lower bounds for arithmetic circuits continues to remain extremely modest.

Most of the research in algebraic complexity theory so far considers arithmetic circuits and multivariate polynomials as *formal* objects and studies the complexity of *syntactic* representation of polynomials over the underlying field. However, in this work, we aim to study the *semantic* or *functional* analogue of the complexity of computing multivariate polynomials. We formally define this notion below and then try to motivate the definition based on our potential applications.

▶ **Definition 1.1** (Functional equivalence)**.** Let $\mathbb{F}$ be any field and let D be a subset of $\mathbb{F}$. We say that two n-variate polynomials P_1 and P_2 in $\mathbb{F}[x_1, x_2, \ldots, x_n]$ are *functionally* equivalent over the domain D^n if

$$\forall \mathbf{x} \in D^n \quad , \quad P_1(\mathbf{x}) = P_2(\mathbf{x}) \, .$$

This definition of functional equivalence naturally extends to the case of arithmetic circuits functionally computing a family of polynomials, as defined below.

▶ **Definition 1.2** (Functional computation)**.** Let $\mathbb{F}$ be any field and let D be a subset of $\mathbb{F}$. A circuit family $\{C_n\}$ is said to functionally compute a family of polynomials $\{P_n\}$ over the domain D^n if

$$\forall n \in \mathbb{N}, \mathbf{x} \in D^n \quad , \quad C_n(\mathbf{x}) = P_n(\mathbf{x}) \, .$$

Having defined functional computation, we will now try to motivate the problem of proving functional lower bounds for arithmetic circuits.

1.1 Motivation

Improved boolean circuit lower bounds

In the late 80s there was some spectacular progress on the question of lower bounds for bounded depth boolean circuits. In particular, Razborov and Smolensky [16, 15] showed exponential lower bounds for constant depth boolean circuits with AND ($\wedge$), OR ($\vee$), Negations ($\neg$) and $\mod p$ gates for a prime p (i.e the class of $\mathsf{AC}^0[\mathsf{p}]$ circuits). However, the question of proving lower bounds for constant depth boolean circuits which also have $\mod q$ gates for a composite q (i.e the class of general ACC^0 circuits) remained wide open. In general, one major obstacle was that the techniques of Razborov and Smolensky failed for composite moduli, and we could not find alternative techniques which were effective for the problem. Although it is widely believed that the majority function should be hard for such circuits, till a few years ago, we did not even know to show that there is such a language in NEXP (the class of problems in nondeterministic exponential time). In a major breakthrough on this question, Williams [17] showed that there is a function in NEXP which requires ACC^0 circuits of superpolynomial size. Along with the result itself, the paper introduced a new proof strategy for showing such lower bounds. However, it still remains wide open to show that there is a function in deterministic exponential time, which requires ACC^0 circuits of superpolynomial size.

One of our main motivations for studying functional lower bounds for arithmetic circuits is the following lemma which shows that such lower bounds in fairly modest set up would imply a separation between $\#\mathsf{P}$ and ACC^0. A formal statement and a simple proof can be found in section 3.

▶ **Lemma 1.3** (Informal). *Let $\mathbb{F}$ be any field of characteristic zero or at least $\exp(\omega(\text{poly}(\log n)))$. Then, a functional lower bound of $\exp(\omega(\text{poly}(\log n)))$ for the permanent of an $n \times n$ matrix over $\{0,1\}^{n^2}$ for depth-4 arithmetic circuits with bottom fan-in $\text{poly}(\log n)$ imply that $\#\mathsf{P} \neq \mathsf{ACC}^0$.*

In fact, we show that something slightly stronger is true. It suffices to prove functional lower bounds for the model of sums of powers of low degree polynomials for the conclusion in Theorem 1.3 to hold.

At this point, there are two possible interpretations of the statement of Theorem 1.3. For an optimist, it provides another approach to proving new lower bounds for ACC^0, while for a pessimist it points to the fact that the functional lower bounds for depth-4 arithmetic circuits could be possibly very challenging. What makes us somewhat optimistic about this strategy is the fact that in the last few years, we seem to have made substantial progress on the question of proving lower bounds for homogeneous depth-4 circuits in the syntactic setting [6, 4, 7, 10]. In particular, even though the depth-4 circuits obtained in the proof of Theorem 1.3 are not homogeneous, an exponential lower bound for sums of powers of low degree polynomials is known in the syntactic set up. Therefore, it makes sense to try and understand if these bounds can be extended to the functional set up as well.

Lower bounds for homogeneous depth-5 circuits

In a recent work by Kumar and Saptharishi [9], it was shown that over constant size finite fields, *average case functional* lower bounds for homogeneous depth-4 circuits implies lower bounds for homogeneous depth-5 circuits[1]. More precisely, the following lemma was shown:

▶ **Lemma 1.4** ([9]). *Let $\mathbb{F}_q$ be a finite field such that $q = O(1)$. Let P be a homogeneous polynomial of degree d in n variables over $\mathbb{F}_q$, which can be computed by a homogeneous depth-5 circuit of size at most $O\left(\exp\left(d^{0.499}\right)\right)$. Then, there exists a homogeneous depth-4 circuit C' of bottom fan-in $O(\sqrt{d})$ and top fan-in at most $O\left(\exp\left(d^{0.499}\right)\right)$ such that*

$$\Pr_{x \in \mathbb{F}_q^n}[P(x) \neq C'(x)] \leq \exp(-\Omega(\sqrt{d}))\,.$$

Informally, the lemma shows that over small finite fields strong enough *average case* functional lower bounds for homogeneous depth-4 arithmetic circuit with bounded bottom fan-in are sufficient to show superpolynomial lower bounds for homogeneous depth-5 circuits. Even though in [9], the authors do not take this route to eventually prove their lower bounds, this connection seems like a strong motivation to study the question of proving functional lower bounds for bounded depth arithmetic circuits.

Functional lower bounds for bounded depth arithmetic circuits

It is immediately clear from the definition that *syntactic* computation implies *functional* computation, but vice-versa may not be necessarily true. In this sense, proving lower bounds for functional computation could be potentially harder than proving lower bounds for syntactic computation. From this point of view, once we have syntactic lower bounds for a certain class of circuits, it seems natural to ask if these bounds can be extended to the functional framework as well. The last few years have witnessed substantial progress on the

[1] In fact, such lower bounds for homogeneous depth-4 circuits with bounded bottom fan-in suffice for this application.

question of proving lower bounds for variants of depth-4 arithmetic circuits, and in this work we explore the question of whether these bounds can be extended to the functional setting.

Applications to proof complexity lower bounds

Functional lower bounds have recently found applications for obtaining lower bounds for algebraic proof systems. In particular, Forbes, Shpilka, Tzameret, and Wigderson [3] have given lower bounds in various algebraic circuit measures for any polynomial agreeing with certain functions of the form $\mathbf{x} \mapsto \frac{1}{p(\mathbf{x})}$, where p is a constant-degree polynomial (which is non-zero on the boolean cube). In particular, they used such lower bounds to obtain lower bounds for the various subclasses of the Ideal Proof System (IPS) of Grochow and Pitassi [5].

In the next section, we explore the connections between syntactic and functional computation in a bit more detail, and discuss why the techniques used in proving syntactic lower bounds do not seem conducive to prove lower bounds in the functional setting. Hence, the problem of proving functional lower bounds might lead us to more techniques for arithmetic circuit lower bounds.

1.2 Functional vs syntactic computation

We now discuss the differences and similarities between functional and syntactic computation in a bit more detail. The following observation is easy to see.

▶ Observation 1.5. The following properties follow from Theorem 1.2:

- Any two polynomials P_1 and P_2 which are syntactically equivalent are also functionally equivalent for every choice of D.
- If two polynomials of individual degrees bounded by d are functionally equivalent over any domain of size at least $d+1$, then they are also syntactically equivalent.
- In particular, any two multilinear polynomials which are functionally equivalent over the hypercube $\{0,1\}^n$ are also syntactically equivalent.

For the rest of the paper, our domain of interest will be $D = \{0,1\}$ and we will be interested in polynomials which are functionally the same over the hypercube $\{0,1\}^n$. For brevity, for the rest of the paper, when we say that two polynomials are functionally equivalent, we mean that the domain is the hypercube. As an additional abuse of notation, when we say that a circuit C is functionally equivalent to a polynomial P, we mean that for every $\mathbf{x} \in \{0,1\}^n$, $C(\mathbf{x}) = P(\mathbf{x})$. Observe that functional equivalence over the hypercube is precisely the same as syntactic equivalence when we work modulo the ideal generated by the polynomials $\{x_i^2 - x_i : i \in [n]\}$. However, we find the functional view easier and more convenient to work with.

At this point, one might ask why is the choice of D as $\{0,1\}$ a natural one? The motivation for studying a domain of size 2 stems from the fact that most of the polynomials for which we have syntactic arithmetic circuit lower bounds, are multilinear. For instance, the permanent (Perm), the determinant (Det), the Nisan-Wigderson polynomials (NW) and the iterated matrix multiplication polynomial (IMM) are known to be hard for many natural classes of arithmetic circuits, homogeneous depth three circuits being one such class. Since for any $D \subseteq \mathbb{F}$ such that $|D| \geq 2$, D^n is an interpolating set for multilinear polynomials, it seems natural to ask if there is a small homogeneous depth three arithmetic circuit which is functionally equivalent to any of these polynomials.

Another reason why $\{0,1\}^n$ seems a natural domain to study functional algebraic computation is due to potential connections to boolean circuit lower bounds. It seems natural to ask if the techniques discovered in the quest for arithmetic circuit lower bounds can be

adapted to say something interesting about questions in boolean circuit complexity. And, Theorem 1.3 seems like an encouraging step in this direction.

1.2.1 Functional lower bounds and partial derivatives

Almost all the bounded depth arithmetic circuit lower bounds so far have been proved using techniques based on the partial derivatives of a polynomial. This includes exponential lower bounds for homogeneous depth-3 circuits [11] and lower bounds for homogeneous depth-4 arithmetic circuits [6, 4, 7, 10]. At a high level, the proofs have the following structure:

- Define a function $\Gamma : \mathbb{F}[\mathbf{x}] \to \mathbb{N}$, called the complexity measure, which serves as an indicator of the hardness of a polynomial.
- For all *small* arithmetic circuits in the model of interest, show that Γ has a non-trivial upper bound.
- For the target hard polynomial, show that Γ is large. Comparing this with the upper bound in step 2 leads to a contradiction if the hard polynomial had a small arithmetic circuit.

The precise measure Γ used in these proofs varies, but they all build upon the the notion of partial derivatives of a polynomial. The idea is to define $\Gamma(P)$ to be the dimension of a linear space of polynomials defined in terms of the partial derivatives of P. In the syntactic set up, if a circuit C computes a polynomial P, then any partial derivative of C must be equivalent to the corresponding partial derivative of P. This observation along with bounds on the dimension of the partial derivative based linear spaces, led to circuit lower bounds.

However, this clearly breaks down in the case when our only guarantee is that the circuit C and the polynomial P agree as functions on all of $\{0,1\}^n$. Apriori, it is not clear if we can say anything meaningful about how the partial derivatives of C and those of P are related to each other. An extreme case of this is the following example. Let the polynomials P and Q be defined as follows:

$$P = \left(\sum_{i=1}^{n} x_i\right)^n$$

and

$$Q = P \mod I_0$$

Here I_0 is the ideal generated by the polynomials $\{x_i^2 - x_i : i \in [n]\}$. The following items follow easily from the definitions:

- $\forall \mathbf{x} \in \{0,1\}^n, P(\mathbf{x}) = Q(\mathbf{x})$.
- The dimension of the span of partial derivatives of P is at most $n+1$.
- The dimension of the span of partial derivatives of Q is at least 2^n. This follows from the fact that the leading monomial of Q is $x_1 \cdot x_2 \cdots x_n$.

So, clearly the dimension of the partial derivatives of two polynomials which are functionally the same over $\{0,1\}^n$ can be wildly different. Thus, it seems tricky to extend the proofs of syntactic lower bounds to the functional setup. Nevertheless, we do manage to get around this obstacle in certain cases as our results in the next section show. Moreover, we also show that a general solution to this question offers a possibility of proving new lower bounds for boolean circuits, that have so far been beyond our reach so far.

1.3 Our results

We now state our main results.

As our first result, we show functional lower bounds for homogeneous[2] depth-3 circuits. In the syntactic setting such lower bounds were first shown by Nisan and Wigderson [11] using the partial derivative of a polynomial as the complexity measure. However, as we discussed in subsubsection 1.2.1, partial derivative based proofs do not extend to the functional setting in a straightforward manner. We get around this obstacle by working with a different but related complexity measure. We now formally state the theorem:

▶ **Theorem 1.6.** *Let $\mathbb{F}$ be any field. There exists a family $\{P_d\}$ of polynomials of degree d in $n = \text{poly}(d)$ variables in VNP such that any $\Sigma\Pi\Sigma$ circuit of formal degree d which is functionally equivalent to P_d over $\{0,1\}^n$ has size at least $\exp\left(\Omega\left(d \log n\right)\right)$.*

As our second result, we show similar functional analogues of the homogeneous depth-4 lower bounds of [7, 10] but under the restriction that the depth-4 circuit computes a polynomial of *low individual degree*. As discussed in the introduction, such lower bounds for depth-4 circuits with bounded bottom fan-in but unbounded individual degree would imply that $\#\mathsf{P} \neq \mathsf{ACC}^0$, and would be a major progress on the question of boolean circuit lower bounds.

▶ **Theorem 1.7.** *Let $\mathbb{F}$ be any field. There exists a family $\{P_d\}$ of polynomials of degree d in $n = \text{poly}(d)$ variables in VNP such that any $\Sigma\Pi\Sigma\Pi$ circuit of formal degree d and individual degree[3] $O(1)$ which is functionally equivalent to P_d over $\{0,1\}^n$ has size at least $\exp\left(\Omega\left(\sqrt{d}\log n\right)\right)$.*

Our techniques for the proof of Theorem 1.7 are again different from the proofs of homogeneous depth-4 lower bounds in the syntactic setting. We introduce a family of new complexity measures, which are functional in their definition (as opposed to partial derivative based measures), and use them to capture functional computation. The family of measures, called *Shifted Evaluation dimension* is a shifted analogue of the well known notion of evaluation dimension, which has had many applications in algebraic complexity (for instance, in multilinear formula, circuit lower bounds [13, 12, 14]). We believe that the measure is of independent interest, and could have other potential applications.

Elementary symmetric polynomials

In their paper [11], Nisan and Wigderson showed an exponential lower bound on the size of homogeneous depth-3 circuits computing the elementary symmetric polynomials. A curious consequence of our proof, is that we are unable to show an analogue of Theorem 1.6 for the elementary symmetric polynomials. One of the reasons for this is the fact that the elementary symmetric polynomials have a *small* evaluation dimension complexity (the complexity measure used for this lower bound), hence our proof technique fails. However, it turns out the at least over fields of sufficiently large characteristic, there are polynomial sized depth-3 circuits of low

[2] Our lower bounds require that the formal degree of the circuit and the degree of the polynomial are *close* to each other. Homogeneity guarantees this condition, but is a much stronger condition than what we need for our proofs to work.

[3] The bounds do not extend to the case of individual degree $\omega(\log n)$, but may still hold for extremely slowly growing functions of n. However, for clarity of presentation, we work with constant individual degree throughout this paper.

formal degree which are functionally equivalent to the elementary symmetric polynomials over $\{0,1\}^n$. The upper bounds are based on the simple observation that for any d and $x \in \{0,1\}^n$, the value of $Sym_d(x)$ (elementary symmetric polynomial of degree d) is equal to $\binom{h(x)}{d}$, where $h(x) = \sum_i x_i$ is the hamming weight of x. In particular, for $d = 1$, the polynomial $\sum_i x_i$ is functionally equivalent to Sym_1, the polynomial $\frac{(\sum_i x_i)(\sum_i x_i - 1)}{2}$ is functionally equivalent to Sym_2 and so on. In particular, there is a polynomial which is a product of d affine forms which is equivalent to Sym_d. However, over fields of low characteristic, the complexity of the elementary symmetric polynomials for functional computation by depth-3 (or even depth-4) circuits is not clear to us and is an interesting open question.

Comparison to Kayal, Saha, Tavenas [8]

In a recent independent result, Kayal, Saha and Tavenas showed exponential lower bounds for depth-4 circuits of bounded individual degree computing an explicit polynomial in VP. Their proof uses a complexity measure called *skew shifted partials* which is very similar in spirit to the notion of *shifted evaluation dimension*, the complexity measure we use. Even though the results seem related, none of them subsumes the other. For our proof, we require that the formal degree of the depth-4 circuit is small (homogeneity), in addition to the individual degree being small, whereas in [8] the authors only require the individual degree of the circuit to be small. In this sense, their result is for a more general model than ours. However, for our lower bounds, we only require the circuit to agree with the target hard polynomial over $\{0,1\}^n$ while the proof in [8] is for syntactically computing the hard polynomial. Hence, the results are incomparable.

1.4 Organization of the paper

We set up some notations to be used in the rest of the paper in section 2. We prove the connections between functional lower bounds for depth-4 circuits and lower bounds for ACC^0 in section 3. We introduce our main complexity measure in section 4. We define and study the properties of the hard polynomials for our lower bounds in section 5. We present the proof of Theorem 1.6 in section 6 and the proof of Theorem 1.7 in section 7.

2 Notation

We now setup some notation to be used for the rest of the paper.

- Throughout the paper, we shall use bold-face letters such as $\mathbf{x}$ to denote a set $\{x_1, \ldots, x_n\}$. Most of the times, the size of this set would be clear from context. We shall also abuse this notation to use $\mathbf{x}^{\mathbf{e}}$ to refer to the monomial $x_1^{e_1} \cdots x_n^{e_n}$.
- The set of formal variables in this paper denoted by $\mathbf{x}$ of size n shall often be partitioned into sets $\mathbf{y}$ and $\mathbf{z}$. We shall use $\mathbf{x} = \mathbf{y} \sqcup \mathbf{z}$ to denote this and use n_y and n_z to denote the sizes of $\mathbf{y}$ and $\mathbf{z}$ respectively.
- For an integer $m > 0$, we shall use $[m]$ to denote the set $\{1, \ldots, m\}$.
- We shall use the short-hand $\partial_{\mathbf{x}^{\mathbf{e}}}(P)$ to denote

$$\frac{\partial^{e_1}}{\partial x_1^{e_1}} \left(\frac{\partial^{e_2}}{\partial x_2^{e_2}} \left(\cdots (P) \cdots \right) \right).$$

- For a set of polynomials $\mathcal{P}$ shall use $\partial_{\mathbf{y}}^{=k}\mathcal{P}$ to denote the set of all k-th order partial derivatives of polynomials in $\mathcal{P}$ with respect to y variables only, and $\partial_{\mathbf{y}}^{\leq k}\mathcal{P}$ similarly.

Also, $\mathbf{x}^{=\ell}\mathcal{P}$ shall refer to the set of polynomials of the form $\mathbf{x}^{\mathbf{e}} \cdot P$ where $\mathsf{Deg}(\mathbf{x}^{\mathbf{e}}) = \ell$ and $P \in \mathcal{P}$. Similarly $\mathbf{x}^{\leq\ell}\mathcal{P}$.

- For a polynomial $P \in \mathbb{F}[\mathbf{x}]$ and for a set $S \subseteq \mathbb{F}^n$, we shall denote by $\mathsf{Eval}_S(P)$ the vector of the evaluation of P on points in S (in some natural predefined order like say the lexicographic order). For a set of vectors V, their span over $\mathbb{F}$ will be denoted by $\mathsf{Span}(V)$ and the dimension of their span by $\mathsf{Dim}(V)$.
- We use $\{0,1\}^n_{\leq k}$ to denote the set of all boolean vectors of length n which have at most k ones.

3 Functional lower bounds for depth-4 circuits and ACC^0

In this section, we show that strong enough functional lower bounds for even very special depth-4 arithmetic circuits are sufficient to imply new lower bounds for ACC^0. The proof follows from a simple application of a well known characterization of ACC^0 by Yao [18] and Beigel and Tarui [2]. The following version of the theorem is from Arora-Barak [1].

▶ **Theorem 3.1** ([18, 2]). *If a function $f : \{0,1\}^n \to \{0,1\}$ is in ACC^0, then f can be computed by a depth 2 circuit with a symmetric gate with quasipolynomial $\left(\exp(\log^{O(1)} n)\right)$ fan-in at the output level and $\vee$ gates with polylogarithmic $\left(\log^{O(1)} n\right)$ fan-in at the bottom level.*

We now prove the following lemma which shows *functional* upper bound for ACC^0.

▶ **Lemma 3.2.** *Let $\mathbb{F}$ be any field of characteristic zero or at least $\exp(\omega(\mathrm{poly}(\log n)))$. If a function $f : \{0,1\}^n \to \{0,1\}$ is in ACC^0, then there exists a polynomial $P_f \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ such that the following are true:*

- *For every $\mathbf{x} \in \{0,1\}^n$, $f(\mathbf{x}) = P_f(\mathbf{x})$.*
- *P_f can be computed by a quasipolynomial sized $\Sigma\wedge\Sigma\Pi$ circuit with bottom fan-in at most $\mathrm{poly}(\log n)$, which are depth-4 circuits where the product gates in the second level[4] are powering gates.*

Proof. From Theorem 3.1, we know that there exists a symmetric function h and multilinear polynomials $g_1, g_2, \ldots, g_t$ such that

- $t = \exp(\mathrm{poly}(\log n))$.
- For every $\mathbf{x} \in \{0,1\}^n$, $f(\mathbf{x}) = h(g_1(\mathbf{x}), g_2(\mathbf{x}), \ldots, g_t(\mathbf{x}))$.
- Each g_i is a multilinear polynomial in at most $\mathrm{poly}(\log n)$ variables.
- For every $\mathbf{x} \in \{0,1\}^n$ and $j \in [t]$, $g_j(\mathbf{x}) \in \{0,1\}$.

From the last item above, we know that the g_is only take boolean values on inputs from $\{0,1\}^n$. Since h is symmetric, it follows that its value on boolean inputs only depends upon the hamming weight of its input. Hence, h is in fact a function of $\sum_{i\in[t]} g_i$. Therefore, over any field of characteristic zero or larger than t, there exists a univariate polynomial P_h of degree at most t over reals, such that

$$\forall \mathbf{x} \in \{0,1\}^n, h\left(g_1(\mathbf{x}), g_2(\mathbf{x}), \ldots, g_t(\mathbf{x})\right) = P_h\left(\sum_{i\in[t]} g_i(\mathbf{x})\right).$$

[4] Throughout this paper, we will assume that our circuits are levelled with alternating $+$ and $\times$ gates. The output gate is level 1 and its inputs are at level 2 and so on.

The lemma now follows from the fact that each g_i is a multilinear polynomial in $\text{poly}(\log n)$ variables. ◀

Theorem 3.2 now immediately implies the following lemma.

▶ **Lemma 3.3.** *Let $\mathbb{F}$ be any field of characteristic zero or at least $\exp(\omega(\text{poly}(\log n)))$. Then, an $\exp(\omega(\text{poly}(\log n)))$ functional lower bound for a function on n variables for $\Sigma \wedge \Sigma\Pi^{[\text{poly}(\log n)]}$ circuits over $\mathbb{F}$ would imply that f is not in ACC^0.*

4 The complexity measure

In the lower bounds for homogeneous depth four circuits [7, 10], the complexity measure used was the *dimension of projected shifted partial derivatives*. The following definition is not the same as used in [7, 10], but this slight variant would be easier to work with for our applications. We abuse notation to call it "projected shifted partial derivatives" as it continues to have the essence of the original definition. A discussion on the precise differences between the following definition and the original definition of [7, 10] is present in Appendix A

▶ **Definition 4.1** (Projected shifted partial derivatives)**.** Let $\mathbf{x} = \mathbf{y} \sqcup \mathbf{z}$ with $|\mathbf{y}| = n_y$ and $|\mathbf{z}| = n_z$, and let S be the set of all strings in $\{0,1\}^{n_y+n_z}$ that are zero on the first n_y coordinates. If k, ℓ are some parameters, the *dimension of projected shifted partial derivatives* for any polynomial $P(\mathbf{y}, \mathbf{z}) \in \mathbb{F}[\mathbf{y}, \mathbf{z}]$, denoted by $\Gamma^{\text{PSPD}}_{k,\ell}(P)$, is defined as

$$\Gamma^{\text{PSPD}}_{k,\ell}(P) \quad := \quad \text{Dim}\left\{\text{Eval}_S\left(\mathbf{z}^{=\ell}\partial^{=k}_{\mathbf{y}}(P)\right)\right\}.$$

The above measure is still syntactic as partial derivatives are not useful in the functional setting. For the functional setting, we shall use a different measure for our lower bound that we call *the shifted evaluation dimension*. We now define the complexity measure that we shall be using to prove the lower bound. For brevity, we shall assume that our set of variables $\mathbf{x}$ is partitioned into $\mathbf{y}$ and $\mathbf{z}$. For our proofs, we shall use a carefully chosen partition. We now formally define the notion of *shifted evaluation dimension* of a polynomial below.

▶ **Definition 4.2** (Shifted evaluation dimension)**.** Let ℓ and k be some parameters and let $\mathbf{x} = \mathbf{y} \sqcup \mathbf{z}$ such that $|\mathbf{y}| = n_y$ and $|\mathbf{z}| = n_z$. For any polynomial $P \in \mathbb{F}[\mathbf{y}, \mathbf{z}]$, define $\Gamma_{k,\ell}(P)$ as

$$\Gamma^{\text{SED}}_{k,\ell}(P) \quad := \quad \text{Dim}\left\{\text{Eval}_{\{0,1\}^{n_z}}\left(\mathbf{z}^{=\ell} \cdot \{P(\mathbf{a}, \mathbf{z}) : \mathbf{a} \in \{0,1\}^{n_y}_{\leq k}\}\right)\right\}.$$

Informally, for every polynomial P, we fix a partition of the input variables into $\mathbf{y}$ and $\mathbf{z}$ and generate a linear space by the following algorithm.

- We take the projections of P obtained by setting each of the y variables to $0, 1$ such that the number of y variables set to 1 is at most k.
- We shift the polynomials obtained in step 1 by all monomials in variables $\mathbf{z}$ of degree ℓ.
- Observe that the polynomials obtained at the end of step two are polynomials only in the $\mathbf{z}$ variables. We now look at the evaluation vectors of these polynomials over $\{0,1\}^{n_z}$.

The complexity measure of the polynomial P is defined as the dimension of the linear space generated by the vectors obtained at the end of step 3 in the algorithm above. For our proof, we will pick a careful partition of the variables $\mathbf{x}$ into $\mathbf{y}$ and $\mathbf{z}$ and look at $\Gamma^{\text{SED}}_{k,\ell}(P)$. The following lemma highlights the key reason of utility of the above measure to functional lower bounds.

▶ **Lemma 4.3** (Functional equivalence and shifted evaluation dimension)**.** *Let $P \in \mathbb{F}[\mathbf{x}]$ and $Q \in \mathbb{F}[\mathbf{x}]$ be any two polynomials which are functionally equivalent over $\{0,1\}^n$. Then, for every choice of k, ℓ and partition $\mathbf{x} = \mathbf{y} \sqcup \mathbf{z}$*

$$\Gamma^{\mathrm{SED}}_{k,\ell}(P) \quad = \quad \Gamma^{\mathrm{SED}}_{k,\ell}(Q)\,.$$

Proof. The proof easily follows from the fact that the measure $\Gamma^{\mathrm{SED}}_{k,\ell}(P)$ is the dimension of a linear space which is generated by vectors which correspond to evaluations of P over subcubes of $\{0,1\}^n$. Hence, it would be the same for any two polynomials which agree as functions over $\{0,1\}^n$. ◀

▶ Remark. Observe that a lemma analogous to Theorem 4.3 is not true in general for partial derivative based measures. And hence, the proofs for syntactic lower bounds which are based on such measures does not immediately carry over to the functional setting.

4.1 Evaluations vs. partial derivatives

In this section, we show that for polynomials of low individual degree, the notion of shifted evaluation dimension can be used as a proxy for the notion of shifted partial derivatives. This is the key observation that drives the proofs of Theorem 1.6 and Theorem 1.7. We first consider the case when the polynomial is *set-multilinear* in which case derivatives can be directly related to careful evaluations.

4.1.1 For set-multilinear polynomials

The explicit polynomials we shall be working with in this paper would be *set-multilinear*. An example to keep in mind is Det_n or Perm_n where the variables can be partitioned into rows and each monomial involves exactly one variable from each part.

▶ **Definition 4.4** (Set-multilinear polynomials)**.** A polynomial P is said to be *set-multilinear* with respect to the a partition $\mathbf{x} = \mathbf{x}_1 \sqcup \cdots \sqcup \mathbf{x}_r$ if every monomial of P involves exactly[5] one variable from each $\mathbf{x}_i$.

We begin with the following simple observation.

▶ Observation 4.5. Let $P \in \mathbb{F}[\mathbf{x}]$ be a set-multilinear with respect to a partition $\mathbf{x} = \mathbf{x}_1 \sqcup \cdots \sqcup \mathbf{x}_r$. Let $\mathbf{y} = \mathbf{x}_1 \cup \cdots \cup \mathbf{x}_k$ for some $k \leq r$ and let $\mathbf{z} = \mathbf{x} \setminus \mathbf{y}$. Then, for any degree k monomial $\mathbf{y}^{\mathbf{e}}$ that is set-multilinear with respect to $\mathbf{x}_1 \sqcup \cdots \sqcup \mathbf{x}_k$, we have

$$\frac{\partial P}{\partial \mathbf{y}^{\mathbf{e}}} \quad = \quad P(\mathbf{e}, \mathbf{z}).$$

Proof. We shall prove this by induction on k. Suppose $\mathbf{y} = \mathbf{x}_1$ and $y_1 \in \mathbf{x}_1$. Since P is set-multilinear, we can write P as

$$P(\mathbf{x}_1, \cdots, \mathbf{x}_r) \quad = \quad \sum_{y_i \in \mathbf{x}_1} y_i \cdot P_i(\mathbf{x}_2, \cdots, \mathbf{x}_r).$$

Hence it follows that $\partial_{y_1}(P)$ equals P_1, which is also the partial evaluation of P where y_1 is set to 1 and all other $y_i \in \mathbf{x}_1$ is set to zero. Hence, if $y_1 = \mathbf{y}^{\mathbf{e}}$, then $\partial_{y_1}(P) = P(\mathbf{e}, \mathbf{x}_2, \cdots, \mathbf{x}_r)$. The claim follows by repeating this argument on $P(\mathbf{e}, \mathbf{x}_2, \cdots, \mathbf{x}_r)$ which continues to be set-multilinear. ◀

[5] sometimes in the literature the word 'exactly' is replaced by 'at most' but in this paper we would be dealing with this definition.

Theorem 4.5 immediately implies the following corollary, which shows that for set-multilinear polynomials *shifted* evaluation dimension and *shifted* partial derivatives are the same quantity if we choose our set of derivatives carefully.

▶ **Corollary 4.6.** *Let $P(\mathbf{x})$ be a set-multilinear polynomial with respect to $\mathbf{x} = \mathbf{x}_1 \sqcup \cdots \sqcup \mathbf{x}_r$. Suppose $\mathbf{y} = \mathbf{x}_1 \cup \cdots \cup \mathbf{x}_k$ and $\mathbf{z} = \mathbf{x} \setminus \mathbf{y}$. Then if we consider the dimension of projected shifted partials with respect to set-multilinear monomials in $\mathbf{y}$, we have*

$$\Gamma^{\mathrm{PSPD}}_{k,\ell}(P) \quad \leq \quad \Gamma^{\mathrm{SED}}_{k,\ell}(P).$$

4.1.2 For low individual degree polynomials

We now proceed to show that an *approximation* of the Theorem 4.6 also holds for polynomials of low individual degree.

▶ **Lemma 4.7.** *Let $P(\mathbf{y}, \mathbf{z})$ be a polynomial with individual degree at most r. Then, for every choice of parameters k and ℓ*

$$\left\{ P(\mathbf{a}, \mathbf{z}) : a \in \{0,1\}^{n_y}_{\leq k} \right\} \quad \subseteq \quad \mathsf{Span}\left(\left(\partial^{\leq rk} P\right)_{\mathbf{y}=\mathbf{0}} \right).$$

Proof. For the rest of this proof, we shall think of P as an element $P_{\mathbf{z}}(\mathbf{y}) \in \mathbb{F}[\mathbf{z}][\mathbf{y}]$. Let $\mathbf{a}$ be any point in $\{0,1\}^{n_y}$. Then by the Taylor's expansion, we know that

$$P_{\mathbf{z}}(\mathbf{y} + \mathbf{a}) \quad = \quad \sum_{\mathbf{e}} \mathbf{a}^{\mathbf{e}} \cdot \partial_{\mathbf{y}^{\mathbf{e}}}(P_{\mathbf{z}})(\mathbf{y}).$$

If the support of $\mathbf{a}$ is at most k, then for every $\mathbf{e}$ such that $\|\mathbf{e}\|_0 > k$, we would have $\mathbf{a}^{\mathbf{e}} = 0$. Moreover, since P is a polynomial of individual degree at most r, it follows that if any coordinate of $\mathbf{e}$ is more than r then

$$\partial_{\mathbf{y}^{\mathbf{e}}}(P_{\mathbf{z}}) = 0.$$

In summary, for any $\mathbf{a}$ such that $\|\mathbf{a}\|_0 \leq k$,

$$\begin{aligned} P_{\mathbf{z}}(\mathbf{y} + \mathbf{a}) \quad &= \quad \sum_{\substack{\mathbf{e}:\|\mathbf{e}\|_0 \leq k, \\ \|\mathbf{e}\|_1 \leq rk}} \mathbf{a}^{\mathbf{e}} \cdot \partial_{\mathbf{y}^{\mathbf{e}}}(P_{\mathbf{z}})(\mathbf{y}) \\ \implies P_{\mathbf{z}}(\mathbf{a}) \quad = \quad P(\mathbf{a}, \mathbf{z}) \quad &= \quad \sum_{\substack{\mathbf{e}:\|\mathbf{e}\|_0 \leq k, \\ \|\mathbf{e}\|_1 \leq rk}} \mathbf{a}^{\mathbf{e}} \cdot \left(\partial_{\mathbf{y}^{\mathbf{e}}}(P_{\mathbf{z}})\right)_{\mathbf{y}=\mathbf{0}} \quad \in \quad \mathsf{Span}\left(\left(\partial^{\leq rk} P\right)_{\mathbf{y}=\mathbf{0}} \right). \end{aligned}$$ ◀

We are now ready to prove our main technical claim of this section.

▶ **Lemma 4.8.** *Let $P(\mathbf{y}, \mathbf{z})$ be a polynomial with individual degree at most r. Then, for every choice of parameters k and ℓ,*

$$\Gamma^{\mathrm{SED}}_{k,\ell}(P) \quad \leq \quad \Gamma^{\mathrm{PSPD}}_{rk,\ell}(P)$$

Proof. From Theorem 4.7, we know that

$$\begin{aligned} \left\{ P(\mathbf{a}, \mathbf{z}) : \mathbf{a} \in \{0,1\}^{n_y}_{\leq k} \right\} \quad &\subseteq \quad \mathsf{Span}\left(\left(\partial^{\leq rk} P\right)_{\mathbf{y}=\mathbf{0}} \right) \\ \implies \left\{ \mathbf{z}^{=\ell} \cdot P(\mathbf{a}, \mathbf{z}) : \mathbf{a} \in \{0,1\}^{n_y}_{\leq k} \right\} \quad &\subseteq \quad \mathsf{Span}\left(\mathbf{z}^{=\ell} \cdot \left(\partial^{\leq rk} P\right)_{\mathbf{y}=\mathbf{0}} \right) \end{aligned}$$

By looking at the evaluation vectors on $\{0,1\}^{n_z}$,

$$\begin{aligned}\left\{\mathsf{Eval}_{\{0,1\}^{n_z}}\left(\mathbf{z}^{=\ell}\cdot P(\mathbf{a},\mathbf{z})\right) : \mathbf{a}\in\{0,1\}^{n_y}_{\leq k}\right\} &\subseteq \mathsf{Span}\left(\mathsf{Eval}_{\{0,1\}^{n_z}}\left(\mathbf{z}^{=\ell}\cdot\left(\partial^{\leq rk}P\right)_{\mathbf{y}=\mathbf{0}}\right)\right)\\ &= \mathsf{Span}\left(\mathsf{Eval}_{\{0\}^{n_y}\times\{0,1\}^{n_z}}\left(\mathbf{z}^{=\ell}\cdot\partial^{\leq rk}P\right)\right)\end{aligned}$$

Taking the dimension of the linear spans on both sides completes the proof. ◀

5 Nisan-Wigderson polynomial families

In this section, we formally define the family of Nisan-Wigderson polynomials and mention some known results about lower bounds on the their projected shifted partials complexity [7, 10, 9]. These bounds will be critically used in our proof.

▶ **Definition 5.1** (Nisan-Wigderson polynomial families)**.** Let d, m, e be arbitrary parameters with m being a power of a prime, and $d, e \leq m$. Since m is a power of a prime, let us identify the set $[m]$ with the field $\mathbb{F}_m$ of m elements. Note that since $d \leq m$, we have that $[d] \subseteq \mathbb{F}_m$. The Nisan-Wigderson polynomial with parameters d, m, e, denoted by $\mathsf{NW}_{d,m,e}$ is defined as

$$\mathsf{NW}_{d,m,e}(\mathbf{x}) \quad = \quad \sum_{\substack{p(t)\in\mathbb{F}_m[t]\\ \mathsf{Deg}(p)<e}} x_{1,p(1)}\cdots x_{d,p(d)}$$

That is, for every univariate polynomial $p(t) \in \mathbb{F}_m[t]$ of degree less that e, we add one monomial that encodes the 'graph' of p on the points $[d]$.

This is a homogeneous, multilinear polynomial of degree d over dm variables with exactly m^e monomials. Furthermore, the polynomial is *set-multilinear* with respect to $\mathbf{x} = \mathbf{x}_1 \sqcup \cdots \sqcup \mathbf{x}_d$ where $\mathbf{x}_i = \{x_{i1}, \cdots, x_{im}\}$.

We now state the following lemma which shows a lower bound on the $\Gamma^{\mathrm{PSPD}}_{k,\ell}(\mathsf{NW}_{d,m,e})$ for an appropriate choice of parameters. We will then use this bound along with Theorem 4.6 to show a lower bound on $\Gamma^{\mathrm{SED}}_{k,\ell}(NW_{d,m,e})$. The lower bound on $\Gamma^{\mathrm{PSPD}}_{k,\ell}(\mathsf{NW}_{d,m,e})$ was shown in two independent proofs by Kayal et al. [7] and by Kumar and Saraf [10]. The version stated below is from a strengthening of these bounds by Kumar and Saptharishi [9].

▶ **Lemma 5.2.** *For every d and $k = O(\sqrt{d})$ there exists parameters m, e, ϵ such that $m = \Theta(d^2)$ and $\epsilon = \Theta\left(\frac{\log d}{\sqrt{d}}\right)$ with*

$$\begin{aligned} m^k &\geq (1+\epsilon)^{2(d-k)}\\ m^{e-k} &= \left(\frac{2}{1+\epsilon}\right)^{d-k}\cdot \mathrm{poly}(m).\end{aligned}$$

For such a choice of parameters, let $\mathbf{x} = \{x_{ij} : i \in [d], j \in [m]\} = \mathbf{x}_1 \sqcup \cdots \sqcup \mathbf{x}_d$ where $\mathbf{x}_i = \{x_{i1}, \ldots, x_{im}\}$. Let $\mathbf{y} = \mathbf{x}_1 \sqcup \cdots \sqcup \mathbf{x}_k$ and $\mathbf{z} = \mathbf{x} \setminus \mathbf{y}$. If ℓ is a parameter that satisfies $\ell = \frac{n_z}{2}(1-\epsilon)$, then over any field $\mathbb{F}$, we have[6]

$$\Gamma^{\mathrm{PSPD}}_{k,\ell}(\mathsf{NW}_{d,m,e}(\mathbf{y},\mathbf{z})) \quad \geq \quad \binom{n_z}{\ell+d-k}\cdot\exp(-O(\log^2 d)).$$

[6] We remark that in the calculations in [7, 10, 9], the shifted monomials consist of both the $\mathbf{y}$ and $\mathbf{z}$ variables, while here we only shift by $\mathbf{z}$ variables. But the calculations still go through since the parameters continue to satisfy the constraints needed for soundness of the calculation.

From Theorem 4.6, we immediately have the following crucial lemma.

▶ **Lemma 5.3.** *Let d, m, e, ℓ be parameters as defined in Theorem 5.2 and let $\mathbf{y}$ and $\mathbf{z}$ be the partition of variables $\mathbf{x}$ as in Theorem 5.2. Then,over any field $\mathbb{F}$, we have*

$$\Gamma^{\mathrm{SED}}_{k,\ell}(\mathsf{NW}_{d,m,e}(\mathbf{y},\mathbf{z})) \quad \geq \quad \binom{n_z}{\ell+d-k} \cdot \exp(-O(\log^2 d)).$$

6 Functional lower bounds for depth-3 circuits

In this section, we complete the proof of Theorem 1.6. We start by defining the exact hard polynomial for which our lower bound is shown.

Hard polynomials for the lower bound

We will prove Theorem 1.6 for the polynomial $\mathsf{NW}_{d,m,e}$ for an appropriate choice of the parameters.

▶ **Lemma 6.1.** *Let the parameters e and d be chosen so that $e = d/2 - 1$, and let $k = e + 1$. Let the variables $\mathbf{x}$ in $\mathsf{NW}_{d,m,e}$ be partitioned into $\mathbf{y} = \{x_{ij} \ : \ i \in [k], j \in [m]\}$ and $\mathbf{z} = \mathbf{x} \setminus \mathbf{y}$. Then*

$$\Gamma^{\mathrm{SED}}_{k,0}(\mathsf{NW}_{d,m,e}(\mathbf{y},\mathbf{z})) \quad \geq \quad m^{d/2}\,.$$

Proof. Let the set of monomials S be defined as

$$S = \left\{ \prod_{i=1}^{k} x_{i,j_i} : j_i \in [m] \right\}.$$

Observe that for every monomial $\mathbf{x}^\alpha$ in S, the partial derivative of $\mathsf{NW}_{d,m,e}$ with respect to $\mathbf{x}^\alpha$, is a monomial in $\mathbf{z}$. This is due to the fact that $e < d/2$ and no two distinct univariate polynomials of degree $d/2$ can agree at more than $d/2$ many points. Moreover for every two distinct monomials $\mathbf{x}^\alpha$ and $\mathbf{x}^\beta$ in S,

$$\frac{\partial \mathsf{NW}_{d,m,e}}{\partial \mathbf{x}^\alpha} \quad \neq \quad \frac{\partial \mathsf{NW}_{d,m,e}}{\partial \mathbf{x}^\beta}\,.$$

Hence,

$$\Gamma^{\mathrm{PSPD}}_{k,0}(\mathsf{NW}_{d,m,e}) = |S| = m^{d/2}\,.$$

Since $\mathsf{NW}_{d,m,e}$ is a set-multilinear with respect to the rows of variable matrix, by Theorem 4.5, it follows that

$$\Gamma^{\mathrm{SED}}_{k,0}(\mathsf{NW}_{d,m,e}) = m^{d/2}\,.$$ ◀

Complexity of the model

▶ **Lemma 6.2.** *The $C(\mathbf{x})$ be a $\Sigma\Pi\Sigma$ circuit of formal degree d and top fan-in s. Then, for all choices of k and any partition of $\mathbf{x}$ into $\mathbf{y}$ and $\mathbf{z}$,*

$$\Gamma^{\mathrm{SED}}_{k,0}(C) \quad \leq \quad s \cdot 2^d\,.$$

Proof. Observe that for any choice of k and ℓ, $\Gamma^{\mathrm{SED}}_{k,\ell}$ is a subadditive measure. Therefore, it is enough to upper bound the value of $\Gamma^{\mathrm{SED}}_{k,0}()$ for every product gate in C by 2^d. Let

$$Q(\mathbf{y},\mathbf{z}) = \prod_{i=1}^{d} L_i$$

be any product gate of formal degree at most d in C. Since each L_i is a linear form, we can express it as $L_i = L_{yi} + L_{zi}$, where L_{yi} and L_{zi} are the parts of L_i consisting entirely of $\mathbf{y}$ and $\mathbf{z}$ variables respectively. Therefore,

$$Q(\mathbf{y},\mathbf{z}) = \sum_{S\subseteq[d]} \prod_{i\in S} L_{yi} \cdot \prod_{j\notin S} L_{zj}\,.$$

Now observe that by

$$\{Q(\mathbf{a},\mathbf{z}) : \mathbf{a}\in\{0,1\}^{n_y}\} \quad\subseteq\quad \mathsf{Span}\left(\left\{\prod_{j\notin S} L_{zj} : S\subseteq[d]\right\}\right)$$

Therefore,

$$\Gamma^{\mathrm{SED}}_{k,0}(C) \quad\leq\quad 2^d\,.$$

The lemma now follows by subadditivity. ◀

Wrapping up the proof

We are now ready to complete the proof of Theorem 1.6.

▶ **Theorem 6.3.** *Let $\mathbb{F}$ be any field, and let d, m, e be parameters such that $e = d/2 - 1$ and $m = \mathrm{poly}(d)$. Let C be a $\Sigma\Pi\Sigma$ circuit of formal degree d which is functionally equivalent to the polynomial $\mathsf{NW}_{d,m,e}$. Then*

$$\mathit{Size}(C) \geq m^{d/2}/2^d\,.$$

Proof. Let $k = e+1$ and consider a partition of variables into $\mathbf{y}$ and $\mathbf{z}$ where all the variables in the first k rows of the variable matrix are labelled $\mathbf{y}$ and the remaining variables are labelled $\mathbf{z}$. Now, the theorem immediately follows from Theorem 6.1 and Theorem 6.2. ◀

7 Functional lower bounds for depth-4 circuits

In this section, we prove Theorem 1.7. We first define the family of polynomials for which our lower bounds apply.

Hard polynomials for the lower bound

For the proof of Theorem 1.7, we would have to show that a statement in the spirit of Theorem 5.3 is also true for a *random projection* of our hard polynomial. Even though we believe[7] that this is true for the polynomial defined in Theorem 5.1, for simplicity, we modify our hard polynomial and in turn prove a lower bound for the following variant of it.

[7] In fact, [7, 10] showed such statements to be true.

▶ **Definition 7.1** (Hard polynomials for the lower bound)**.** Let d, m, e be parameters as defined in Theorem 5.1. Let $p = p(m, d)$ be a parameter and let

$$t = \frac{dm}{p}\,.$$

The polynomial $\mathsf{NW} \circ \mathsf{Lin}$ is defined as

$$\mathsf{NW} \circ \mathsf{Lin}_{d,m,e,p} = \mathsf{NW}_{d,m,e}\left(L(x_{1,1}), L(x_{1,2}), \ldots, L(x_{d,m})\right)$$

where for each $i \in [d], j \in [m]$, $L(x_{i,j})$ is defined as

$$L(x_{i,j}) = \sum_{u=1}^{t} x_{i,j,u}\,.$$

For the rest of this proof, we set $p = (md)^{-0.1}$, and for brevity, we will indicate $\mathsf{NW} \circ \mathsf{Lin}_{d,m,e,(md)^{0.1}}$ by $\mathsf{NW} \circ \mathsf{Lin}_{d,m,e}$. Observe that setting p sets t to be equal to $(md)^{1.1}$. We conclude this section with the next lemma where we show that $\mathsf{NW} \circ \mathsf{Lin}_{d,m,e}$ is *robust* under random restrictions where every variable is kept alive with a probability p.

▶ **Lemma 7.2.** *Let p and t be as stated above and let $n = dm$. Let P be a random projection of $\mathsf{NW} \circ \mathsf{Lin}$ obtained by setting every variable in $\{x_{i,j,h} : i \in [d], j \in [m], h \in [t]\}$ to zero with a probability equal to $1 - p$. Then, with a probability at least $1 - o(1)$, $\mathsf{NW}_{d,m,e}$ is a projection of P.*

Proof. For every $i \in [d]$, $j \in [m]$, define the set $A_{i,j}$ as

$$A_{ij} = \{x_{i,j,h} : h \in [t]\}\,.$$

When every variable is being set to zero with a probability $1 - p$, the probability that there exists an $i \in [d]$ and $j \in [m]$ such that all the variables in the set $A_{i,j}$ are set to zero is at most $dm(1-p)^t$. For $p = n^{-0.1}$, the probability is at most $n(1 - n^{-0.1})^{n^{1.1}}$ which is $\exp(-\Omega(n))$.

Therefore, with a probability at least $1 - \exp(-\Omega(n))$, each of the set $A_{i,j}$ has at least one variable alive in P. Now, we set all but one of them to zero for each i, j. Observe that the resulting projection of P is precisely $\mathsf{NW}_{d,m,e}$ up to a relabelling of variables. This proves the lemma. ◀

It should be noted that the polynomial $\mathsf{NW} \circ \mathsf{Lin}$ continues to remain set-multilinear with respect to he rows of the variable matrix.

Upper bound on the complexity of the model

We now show the upper bound on $\Gamma^{\mathrm{SED}}_{k,\ell}(C)$ when C is a depth-4 circuit of individual degree at most r and bottom support s. We will use the following upper bound on $\Gamma^{\mathrm{PSPD}}_{k,\ell}(C)$ from [7, 10].

▶ **Lemma 7.3.** *Let $C(\mathbf{y}, \mathbf{z})$ be a depth-4 circuit, of formal degree at most d and bottom support at most s. Let k and ℓ be parameters satisfying $\ell + ks < n_z/2$. Then*

$$\Gamma^{\mathrm{PSPD}}_{k,\ell}(C) \quad \leq \quad \mathit{Size}(C) \cdot \binom{O\left(\frac{d}{s}\right) + k}{k} \cdot \binom{n_z}{\ell + ks} \cdot \mathrm{poly}(n)\,.$$

The following lemma now immediately follows from Theorem 7.3 and Theorem 4.8.

▶ **Lemma 7.4.** *Let $C(\mathbf{y}, \mathbf{z})$ be a depth-4 circuit, of formal degree at most d, individual degree at most r and bottom support at most s. Let k and ℓ be parameters satisfying $\ell + krs < n_z/2$. Then*

$$\Gamma^{\mathrm{SED}}_{k,\ell}(C) \quad \leq \quad \mathit{Size}(C) \cdot \binom{O\left(\frac{d}{s}\right) + kr}{kr} \cdot \binom{n_z}{\ell + krs} \cdot \mathrm{poly}(n_z).$$

Wrapping up the proof

▶ **Theorem 7.5.** *Let d, m, e be parameters as defined in Theorem 5.2. Let C be a $\Sigma\Pi\Sigma\Pi$ circuit C of formal degree d and individual degree at most $r = O(1)$ over any field $\mathbb{F}$ such that C is functionally equivalent to $\mathsf{NW} \circ \mathsf{Lin}_{d,m,e}$. Then,*

$$\mathit{Size}(C) \geq \exp\left(\Omega\left(\sqrt{d}\log dm\right)\right).$$

Proof. If the size of C is larger than $\exp\left(\frac{\sqrt{d}\log dm}{1000r}\right)$, then we are already done, else the size of C is at most $\exp\left(\frac{\sqrt{d}\log dm}{1000r}\right)$. Let us set every variable in C and $\mathsf{NW} \circ \mathsf{Lin}_{d,m,e}$ to zero independently with a probability $1 - (md)^{-0.1}$. The following claim easily follows via a standard application of the union bound.

▶ **Claim 7.6.** *With probability at least $1 - o(1)$ over the random restrictions as defined above, every product gate at the bottom level of C with support at least $\frac{\sqrt{d}}{100r}$ is set to zero.*

From the above claim and from Theorem 7.2, it follows that there is a $\Sigma\Pi\Sigma\Pi$ circuit C' of formal degree d over $\mathbb{F}$ which is functionally equivalent to $\mathsf{NW}_{d,m,e}$. Let us relabel the variables as $\mathbf{y}$ and $\mathbf{z}$ as described in Theorem 5.2. Let $k = \sqrt{d}$ and let $\ell = \frac{n_z}{2} \cdot (1 - \epsilon)$ where $\epsilon = O\left(\frac{\log d}{\sqrt{d}}\right)$ to be chosen shortly. By Theorem 5.3, we know that for this choice of k and ℓ

$$\begin{aligned}
\Gamma^{\mathrm{SED}}_{k,\ell}(\mathsf{NW}_{d,m,e}(\mathbf{y}, \mathbf{z})) \quad &\geq \quad \binom{n_z}{\ell + d - k} \cdot \exp(-O(\log^2 d)) \\
&\geq \quad \binom{n_z}{\ell} \cdot (1+\epsilon)^{2d-2k} \cdot \exp(-O(\log^2 d))
\end{aligned}$$

Moreover, by Theorem 7.4, we know that

$$\begin{aligned}
\Gamma^{\mathrm{SED}}_{k,\ell}(C') \quad &\leq \quad (dm)^{\sqrt{d}/1000r} \cdot \binom{O\left(\frac{\sqrt{d}}{r}\right) + kr}{kr} \cdot \binom{n_z}{\ell + k \cdot r \cdot \frac{\sqrt{d}}{100r}} \cdot \mathrm{poly}(n_z) \\
&\leq \quad (dm)^{\sqrt{d}/1000r} \cdot 2^{O(\sqrt{d})} \cdot \binom{n_z}{\ell} \cdot (1+\epsilon)^{\frac{d}{50}} \cdot \exp(O(\log^2 d)) \\
&\leq \quad \exp\left(\sqrt{d}\log d/100r\right) \cdot 2^{O(\sqrt{d})} \cdot \binom{n_z}{\ell} \cdot (1+\epsilon)^{\frac{d}{50}} \cdot \exp(O(\log^2 d))
\end{aligned}$$

Now, observe that there exists a constant c such that if ϵ is set to $\frac{c\log d}{\sqrt{d}}$, then

$$\Gamma^{\mathrm{SED}}_{k,\ell}(\mathsf{NW}_{d,m,e}) > \Gamma^{\mathrm{SED}}_{k,\ell}(C').$$

But this is a contradiction since C' computes $\mathsf{NW}_{d,m,e}$. This completes the proof. ◀

8 Open problems

We end with some open questions:

- The main challenge would be to improve Theorem 1.7, and prove it for the model of sums of powers of low degree polynomials. It is not clear to us if the complexity measure used in this paper would be useful.
- The functional lower bounds proved in this paper are for *exact* functional computation. We believe that some of these bounds should also hold in the average case, where the circuit and the polynomial agree on a random point on $\{0,1\}^n$ with a high probability. It is not clear to us if the proof techniques in this paper can be adapted to say something in the average case setting. The most natural attempt to generalize the proofs seem to hit a *matrix rigidity* like obstacle.

Acknowledgements. Part of this work was done while the third author was visiting Rutgers. We are grateful to Eric Allender and DIMACS for funding the visit. We are also grateful to Pravesh Kothari and Madhu Sudan for many helpful conversations.

References

1 Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, New York, NY, USA, 1st edition, 2009.

2 Richard Beigel and Jun Tarui. On acc. *Computational Complexity*, 4(4):350–366, 1994. doi:10.1007/BF01263423.

3 Michael A. Forbes, Amir Shpilka, Iddo Tzameret, and Avi Wigderson. Proof complexity lower bounds from algebraic circuit complexity. Manuscript, 2015.

4 Hervé Fournier, Nutan Limaye, Guillaume Malod, and Srikanth Srinivasan. Lower bounds for depth 4 formulas computing iterated matrix multiplication. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC 2014)*, pages 128–135, 2014. doi:10.1145/2591796.2591824.

5 Joshua A. Grochow and Toniann Pitassi. Circuit complexity, proof complexity, and polynomial identity testing. In *Proceedings of the 55th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2014)*, pages 110–119, 2014. Full version at arXiv:abs/1404.3820. doi:10.1109/FOCS.2014.20.

6 Ankit Gupta, Pritish Kamath, Neeraj Kayal, and Ramprasad Saptharishi. Approaching the chasm at depth four. *Journal of the ACM*, 61(6):33:1–33:16, 2014. Preliminary version in the *28th Annual IEEE Conference on Computational Complexity (CCC 2013)*. doi:10.1145/2629541.

7 Neeraj Kayal, Nutan Limaye, Chandan Saha, and Srikanth Srinivasan. An Exponential Lower Bound for Homogeneous Depth Four Arithmetic Circuits. In *Proceedings of the 55th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2014)*, 2014. doi:10.1109/FOCS.2014.15.

8 Neeraj Kayal, Chandan Saha, and Sébastien Tavenas. On the size of homogeneous and of depth four formulas with low individual degree. *Electronic Colloquium on Computational Complexity (ECCC)*, 2015. eccc:TR15-181. URL: http://eccc.hpi-web.de/report/2015/181/.

9 Mrinal Kumar and Ramprasad Saptharishi. An exponential lower bound for homogeneous depth-5 circuits over finite fields. *Electronic Colloquium on Computational Complexity (ECCC)*, 2015. eccc:TR15-109. URL: http://eccc.hpi-web.de/report/2015/109/.

10 Mrinal Kumar and Shubhangi Saraf. On the power of homogeneous depth 4 arithmetic circuits. In *Proceedings of the 55th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2014)*, 2014. doi:10.1109/FOCS.2014.46.

11 Noam Nisan and Avi Wigderson. Lower bounds on arithmetic circuits via partial derivatives. *Computational Complexity*, 6(3):217–234, 1997. doi:10.1007/BF01294256.

12 Ran Raz. Separation of multilinear circuit and formula size. *Theory of Computing*, 2(1):121–135, 2006. Preliminary version in the *45th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2004)*. doi:10.4086/toc.2006.v002a006.

13 Ran Raz. Multi-linear formulas for permanent and determinant are of super-polynomial size. *Journal of the ACM*, 56(2), 2009. Preliminary version in the *36th Annual ACM Symposium on Theory of Computing (STOC 2004)*. doi:10.1145/1502793.1502797.

14 Ran Raz and Amir Yehudayoff. Lower bounds and separations for constant depth multilinear circuits. *Computational Complexity*, 18(2):171–207, 2009. Preliminary version in the *23rd Annual IEEE Conference on Computational Complexity (CCC 2008)*. doi:10.1007/s00037-009-0270-8.

15 Alexander A. Razborov. Lower bounds on the size of bounded depth circuits over a complete basis with logical addition. *Mathematical notes of the Academy of Sciences of the USSR*, 41(4):333–338, 1987. doi:10.1007/BF01137685.

16 Roman Smolensky. Algebraic methods in the theory of lower bounds for boolean circuit complexity. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing (STOC 1987)*, pages 77–82, 1987. doi:10.1145/28395.28404.

17 Ryan Williams. Non-uniform ACC Circuit Lower Bounds. In *Proceedings of the 26th Annual IEEE Conference on Computational Complexity (CCC 2011)*, pages 115–125, 2011.

18 Andrew Chi-Chih Yao. Separating the polynomial-time hierarchy by oracles. In *Proceedings of the 26th Annual IEEE Symposium on Foundations of Computer Science (FOCS 1985)*, pages 1–10, Oct 1985. doi:10.1109/SFCS.1985.49.

A The evaluation perspective on projected shifted partial derivatives

The notion of projected shifted partial derivatives was first introduced by Kayal, Limaye, Saha and Srinivasan [7] in proving lower bounds for homogeneous depth-4 circuits. The following is the precise definition they used.

▶ **Definition A.1** (Projected shifted partial derivatives of [7])**.** Let k and ℓ be some parameters. The projected shifted partial derivatives of a polynomial $P(\mathbf{y}, \mathbf{z})$, denoted by $\Gamma_{k,\ell}^{\mathrm{PSPD}_0}(P)$, is defined as

$$\Gamma_{k,\ell}^{\mathrm{PSPD}_0}(P) \quad := \quad \mathsf{Dim}\left\{\mathrm{mult}\left(\mathbf{z}^{=\ell}\partial_{\mathbf{y}}^{=k}(P)\right)\right\}$$

where $\mathrm{mult}(f)$ is just the vector of coefficients of all *multilinear* monomials in f in a fixed predefined order.

An alternate way to interpret the above definition is to consider the shifted partial derivatives of P, and *reduce* them under the relation $x_i^2 = 0$, and only then list the coefficients of the surviving monomials. The rationale for this in [7] was to ensure that non-multilinear terms do not interact with multilinear terms in the shifted partial derivatives of P. Hence,

$$\Gamma_{k,\ell}^{\mathrm{PSPD}_0}(P) \quad = \quad \mathsf{Dim}\left\{\mathbf{z}^{=\ell}\partial_{\mathbf{y}}^{=k}(P) \mod \left\{x_i^2 \ : \ i \in [n]\right\}\right\}.$$

Another equally useful definition, which was also employed by Kumar and Saptharishi [9], is to reduce the shifted partial derivatives of P with respect to $x_i^2 = x_i$ instead. This also in

essence ensures that non-multilinear terms do not interact with the relevant multilinear terms by reducing their degree. We shall denote this by $\Gamma^{\mathrm{PSPD}_1}_{k,\ell}(P)$, which is formally defined to be

$$\Gamma^{\mathrm{PSPD}_1}_{k,\ell}(P) \quad := \quad \mathsf{Dim}\left\{\mathbf{z}^{=\ell}\partial^{=k}_{\mathbf{y}}(P) \mod \left\{(x_i^2 - x_i) \ : \ i \in [n]\right\}\right\}.$$

Since any polynomial f has a unique multilinear representation modulo $\left\{x_i^2 - x_i \ : \ i \in [n]\right\}$, it follows that its evaluations on $\{0,1\}^n$ completely determine the coefficients of the reduced polynomial $f \mod \left\{x_i^2 - x_i \ : \ i \in [n]\right\}$. Therefore, if $\Gamma^{\mathrm{PSPD}}_{k,\ell}(P)$ is defined as

$$\Gamma^{\mathrm{PSPD}_2}_{k,\ell}(P) \quad := \quad \mathsf{Dim}\left\{\mathsf{Eval}_{\{0,1\}^n}\left(\mathbf{z}^{=\ell}\partial^{=k}_{\mathbf{y}}(P)\right)\right\},$$

then it follows that

$$\Gamma^{\mathrm{PSPD}_2}_{k,\ell}(P) \quad = \quad \Gamma^{\mathrm{PSPD}_1}_{k,\ell}(P).$$

Finally, if P was set-multilinear with respect to $\mathbf{x} = \mathbf{x}_1 \sqcup \cdots \sqcup \mathbf{x}_r$ and $\mathbf{y} = \mathbf{x}_1 \sqcup \cdots \sqcup \mathbf{x}_k$, then all partial derivatives of order k with respect to $\mathbf{y}$ would be result in polynomials only in $\mathbf{z}$. Therefore for such set-multilinear polynomials,

$$\begin{aligned}\Gamma^{\mathrm{PSPD}_2}_{k,\ell}(P) &= \mathsf{Dim}\left\{\mathsf{Eval}_{\{0,1\}^n}\left(\mathbf{z}^{=\ell}\partial^{=k}_{\mathbf{y}}(P)\right)\right\} \\ &= \mathsf{Dim}\left\{\mathsf{Eval}_{\{0\}^{n_y}\times\{0,1\}^{n_z}}\left(\mathbf{z}^{=\ell}\partial^{=k}_{\mathbf{y}}(P)\right)\right\} \\ &=: \Gamma^{\mathrm{PSPD}}_{k,\ell}(P) \quad \text{as defined in Theorem 4.1.}\end{aligned}$$

The explicit polynomials for which we shall be show the lower bounds would indeed be set-multilinear and hence there is no loss incurred in restricting to only evaluations on $\{0\}^{n_y} \times \{0,1\}^{n_z}$.

For polynomials that are not set-multilinear, clearly

$$\begin{aligned}\Gamma^{\mathrm{PSPD}_2}_{k,\ell}(P) &= \mathsf{Dim}\left\{\mathsf{Eval}_{\{0,1\}^n}\left(\mathbf{z}^{=\ell}\partial^{=k}_{\mathbf{y}}(P)\right)\right\} \\ &\geq \mathsf{Dim}\left\{\mathsf{Eval}_{\{0\}^{n_y}\times\{0,1\}^{n_z}}\left(\mathbf{z}^{=\ell}\partial^{=k}_{\mathbf{y}}(P)\right)\right\} \quad =: \quad \Gamma^{\mathrm{PSPD}}_{k,\ell}(P).\end{aligned}$$

Hence for the purposes of upper-bounding $\Gamma^{\mathrm{PSPD}}_{k,\ell}()$ for say a term in the circuit computing P, taking fewer evaluations only helps.

Arithmetic Circuits with Locally Low Algebraic Rank

Mrinal Kumar[*1] **and Shubhangi Saraf**[†2]

1 **Department of Computer Science, Rutgers University, New Brunswick, USA**
mrinal.kumar@rutgers.edu

2 **Department of Computer Science and Department of Mathematics, Rutgers University, New Brunswick, USA**
shubhangi.saraf@gmail.com

Abstract

In recent years there has been a flurry of activity proving lower bounds for homogeneous depth-4 arithmetic circuits [14, 11, 18, 27], which has brought us very close to statements that are known to imply VP $\neq$ VNP. It is a big question to go beyond homogeneity, and in this paper we make progress towards this by considering depth-4 circuits of *low algebraic rank*, which are a natural extension of homogeneous depth-4 arithmetic circuits.

A depth-4 circuit is a representation of an N-variate, degree n polynomial P as $P = \sum_{i=1}^{T} Q_{i1} \cdot Q_{i2} \cdot \cdots Q_{it}$ where the Q_{ij} are given by their monomial expansion. Homogeneity adds the constraint that for every $i \in [T]$, $\sum_j \text{degree}(Q_{ij}) = n$. We study an extension where, for every $i \in [T]$, the *algebraic rank* of the set of polynomials $\{Q_{i1}, Q_{i2}, \ldots, Q_{it}\}$ is at most some parameter k. We call this the class of $\Sigma\Pi^{(k)}\Sigma\Pi$ circuits. Already for $k = n$, these circuits are a strong generalization of the class of homogeneous depth-4 circuits, where in particular $t \leq n$ (and hence $k \leq n$).

We study lower bounds and polynomial identity tests for such circuits and prove the following results.

1. **Lower bounds:** We give an explicit family of polynomials $\{P_n\}$ of degree n in $N = n^{O(1)}$ variables in VNP, such that any $\Sigma\Pi^{(n)}\Sigma\Pi$ circuit computing P_n has size at least $\exp\left(\Omega(\sqrt{n}\log N)\right)$. This strengthens and unifies two lines of work: it generalizes the recent exponential lower bounds for *homogeneous* depth-4 circuits [18, 27] as well as the Jacobian based lower bounds of Agrawal et al. [2] which worked for $\Sigma\Pi^{(k)}\Sigma\Pi$ circuits in the restricted setting where $T \cdot k \leq n$.
2. **Hitting sets:** Let $\Sigma\Pi^{(k)}\Sigma\Pi^{[d]}$ be the class of $\Sigma\Pi^{(k)}\Sigma\Pi$ circuits with bottom fan-in at most d. We show that if d and k are at most $\text{poly}(\log N)$, then there is an explicit hitting set for $\Sigma\Pi^{(k)}\Sigma\Pi^{[d]}$ circuits of size quasipolynomial in N and the size of the circuit. This strengthens a result of Forbes [8] which showed such quasipolynomial sized hitting sets in the setting where d and t are at most $\text{poly}(\log N)$.

A key technical ingredient of the proofs is a result which states that over any field of characteristic zero (or sufficiently large characteristic), up to a translation, every polynomial in a set of algebraically dependent polynomials can be written as a function of the polynomials in the transcendence basis. We believe this may be of independent interest. We combine this with shifted partial derivative based methods to obtain our final results.

1998 ACM Subject Classification F.2.1 Numerical Algorithms and Problems, I.1.1 Expressions and Their Representation

Keywords and phrases algebraic independence, arithmetic circuits, lower bounds

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.34

[*] Research supported in part by NSF grant CCF-1350572 and by Simons Graduate Fellowship.
[†] Research supported by NSF grant CCF-1350572.

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 34; pp. 34:1–34:27

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

1 Introduction

Arithmetic circuits are natural algebraic analogs of boolean circuits, with the logical operations being replaced by sum and product operations over the underlying field. Valiant [42] developed the complexity theory for algebraic computation via arithmetic circuits and defined the complexity classes VP and VNP as the algebraic analogs of complexity classes P and NP respectively. We refer the interested reader to the survey by Shpilka and Yehudayoff [37] for more on arithmetic circuits.

Two of the most fundamental questions in the study of algebraic computation are the questions of *polynomial identity testing* (PIT)[1] and the question of proving *lower bounds* for explicit polynomials. It was shown by structural results known as *depth reductions* [1, 24, 41] that strong enough lower bounds or PIT results for just (homogeneous) depth-4 circuits, would lead to superpolynomial lower bounds and derandomized PIT for general circuits too. Consequently, depth-4 arithmetic circuits have been the focus of much investigation in the last few years.

Just in the last few years we have seen rapid progress in proving lower bounds for homogeneous depth-4 arithmetic circuits, starting with the work of Gupta et al. [14] who proved exponential lower bounds for homogeneous depth-4 circuits with bounded bottom fan-in and terminating with the results in [18, 27] which showed exponential lower bounds for general homogeneous depth-4 circuits. Any asymptotic improvement in the exponent of these lower bounds would lead to superpolynomial lower bounds for general arithmetic circuits[2]. Most of this progress was based on an understanding of the complexity measure of the family of *shifted partial derivatives* of a polynomial (this measure was introduced in [21]), and other closely related measures.

Although we now know how to use these measure to prove such strong lower bounds for homogeneous depth 4 circuits, the best known lower bounds for non-homogeneous depth three circuits over fields of characteristic zero are just quadratic [36, 38], and those for non-homogeneous depth-4 circuits over any field except $\mathbb{F}_2$ are just about superlinear [30]. It remains an extremely interesting question to get improved lower bounds for these circuit classes.

In sharp contrast to this state of knowledge on lower bounds, the problem of polynomial identity testing is very poorly understood even for depth three circuits. Till a few years ago, almost all the PIT algorithms known were for extremely restricted classes of circuits and were based on diverse proof techniques (for instance, [5, 20, 16, 19, 15, 34, 35, 33, 2, 9, 10, 4]). The work of [2] gave a unified proof of several of them.

It is a big question to go *beyond homogeneity* (especially for proving lower bounds) and in this paper we make progress towards this question by considering depth-4 circuits of *low algebraic rank*, which are a natural extension of homogenous depth-4 arithmetic circuits.

A depth-4 circuit is a respresentation of an N-variate, degree n polynomial P as

$$P = \sum_{i=1}^{T} Q_{i1} \cdot Q_{i2} \cdot \cdots Q_{it}$$

where the Q_{ij} are given by their monomial expansion. Homogeneity adds the constraint that for every $i \in [T]$, $\sum_j \text{degree}(Q_{ij}) = n$. We study an extension where, for every $i \in [T]$,

[1] Given an arithmetic circuit, the problem is to decide if it computes the identically zero polynomial. In the whitebox set up, we are allowed to look inside the wirings of the circuit, while in the blackbox setting, we can only query the circuit at some points.

[2] We refer the interested reader to the surveys of recent lower bounds results by Saptharishi [32, 31]

the *algebraic rank* of the set of polynomials $\{Q_{i1}, Q_{i2}, \ldots, Q_{it}\}$ is at most some parameter k. We call this the class of $\Sigma\Pi^{(k)}\Sigma\Pi$ circuits. Already for $k = n$, these circuits are a strong generalization of the class of homogeneous depth-4 circuits, where in particular $t \leq n$ (and hence $k \leq n$).

We prove exponential lower bounds for $\Sigma\Pi^{(k)}\Sigma\Pi$ circuits for $k \leq n$ and give quasipolynomial time deterministic polynomial identity tests for $\Sigma\Pi^{(k)}\Sigma\Pi$ circuits when k and the bottom fan-in are bounded by poly$(\log N)$. All our results actually hold for a more general class of circuits, where the product gates at the second level can be replaced by an arbitrary circuits whose inputs are polynomials of algebraic rank at most k. In particular, our results hold for representations of a polyonomial P as

$$P = \sum_{i=1}^{T} C_i \left(Q_{i1}, Q_{i2}, \ldots, Q_{it}\right)$$

where, for every $i \in [T]$, C_i is an arbitrary polynomial function of t inputs, and the algebraic rank of the set of polynomials $\{Q_{i1}, Q_{i2}, \ldots, Q_{it}\}$ is at most some parameter k.

1.1 Some background and motivation

Before we more formally define the model and state our results, we give some background and motivation for studying this class of circuits.

Strengthening of the model of homogenous depth-4 circuits

As already mentioned, we know very strong exponential lower bounds for homogenous depth-4 arithmetic circuits. In contrast, for general (non-homogenous) depth-4 circuits, we know only barely superlinear lower bounds, and it is a challenge to obtain improved bounds. $\Sigma\Pi^{(k)}\Sigma\Pi$ circuits with k as large as n (the degree of the polynomial being computed), which is the class we study in this paper, is already a significant strengthening of the model of homogenous depth-4 circuits (since the intermediate degrees could be exponentially large). We provide exponential lower bounds for this model. Note that when $k = N$, $\Sigma\Pi^{(k)}\Sigma\Pi$ circuits would capture general depth-4 arithmetic circuits.

Low algebraic rank and lower bounds

In a recent work, Agrawal et al. [2] studied the notion of circuits of low algebraic rank and by using the Jacobian to capture the notion of algebraic independence, they were able to show exponential lower bounds for a certain class of arithmetic circuits[3]. They showed that over fields of characteristic zero, for any set of polynomials $\{Q_1, Q_2, \ldots, Q_t\}$ of sparsity at most s and algebraic rank k, any arithmetic circuit of the form $C(Q_1, Q_2, \ldots, Q_t)$ which computes the determinant polynomial for an $n \times n$ symbolic matrix must $s \geq \exp(n/k)$. In particular, if $k = \Omega(n)$, then the lower bound becomes trivial. The lower bounds in this paper strengthen this work in two ways.

(1) Our lower bounds hold for a much richer class of circuits. In the model considered by [2], one imposes a global upper bound k on the rank of all the Q_is feeding into some polynomial C. In our model, we can take exponentially many different sets of Q_is each with bounded rank, and apply some polynomial function to each of them and then take a sum.

[3] Even more significantly they also give efficient PIT algorithms for the same class of circuits.

(2) Our lower bounds are stronger - we obtain exponential lower bounds even when k is as large as the degree of the polynomial being computed.

Algebraic rank and going beyond homogeneity

Even though we know exponential lower bounds for homogeneous[4] depth-4 circuits, the best known lower bounds for non-homogeneous depth-4 circuits are barely superlinear [30].

In [13, 12, 36], Grigoriev-Karpinski, Grigoriev-Razborov and Shpilka-Wigderson outlined a program based on “rank" to prove lower bounds for arithmetic circuits. They used the notion of “linear rank" and used it to prove lower bounds for depth-3 arithmetic circuits in the following way: Let $C = \sum_{i=1}^{T} \prod_{j=1}^{t} L_{ij}$ be a depth three (possibly nonhomogeneous) circuit computing a polynomial P of degree n. Now, partition the inputs to the top sum gate to two halves, C_1 and C_2 based on the rank of the inputs feeding into it in the following way. For each $i \in [T]$, if the linear rank of the set of polynomials $\{L_{ij} : j \in [t]\}$ is at most k (for some threshold k), then include the gate i into the sum C_1, else include it into C_2. Therefore,

$$C = C_1 + C_2.$$

Their program had two steps. (1) Show that the subcircuit C_1 is *weak* with respect to some complexity measure, and thus show a lower bound for C_1 (and hence C) when C_2 is trivial. (2) Also since C_2 is“high rank", show that there are many inputs for which C_2 is identically zero. Then try to look at restrictions over which C_2 is identically zero, and show that the lower bounds for C_1 continue to hold.

The following is the natural generalization of this approach to proving lower bounds for depth-4 circuits. Let $C = \sum_{i=1}^{T} \prod_{j=1}^{t} Q_{ij}$ be a depth-4 circuit computing a polynomial P of degree n. Note that in general, the formal degree of C could be much larger than n. Now, we partition the inputs to the top sum gate to two halves, C_1 and C_2 based on the *algebraic rank* of the inputs feeding into it in the following way. For each $i \in [T]$, if the algebraic rank of the set of polynomials $\{Q_{ij} : j \in [t]\}$ is at most k (for some threshold k), then we include the gate i into the sum C_1 else we include it into C_2. Therefore,

$$C = C_1 + C_2.$$

To implement the G-K, G-R and S-W program, as a first step one would show that the subcircuit C_1 is *weak* with respect to some complexity measure, and thus show a lower bound for C_1 (and hence C) when C_2 is trivial. The second step would be to try to look at restrictions over which C_2 is identically zero, and show that the lower bounds for C_1 continue to hold.

For the case of depth-4 circuits, even the first step of showing lower bounds when C_2 is trivial was not known prior to this work (even for $k = 2$). Our results in this paper are an implementation of this first step, as we show exponential lower bounds when the algebraic rank of inputs into each of the product gates is at most n (the degree of the polynomial being computed).

Connections to divisibility testing

Recently, Forbes [8] showed that given two sparse multivariate polynomials P and Q, the question of deciding if P divides Q can be reduced to the question of polynomial identity

[4] These results, infact hold for depth-4 circuits with not-too-large formal degree.

testing for $\Sigma\Pi^{(2)}\Sigma\Pi$ circuits. This question was one of the original motivations for this paper. Although we are unable to answer this question in general, we make some progress towards it by giving a quasipolynomial identity tests for $\Sigma\Pi^{(k)}\Sigma\Pi$ circuits when the various Q_{ij} feeding into the circuit have degree bounded by $\text{poly}(\log N)$ (and we are also able to handle k as large as $\text{poly}(\log N)$.)

Low algebraic rank and PIT

Two very interesting PIT results which are also very relevant to the results in this paper are those of Beecken et al. [3] and those of Agrawal et al. [2]. The key idea explored in both these papers is that of algebraic independence. Together, they imply efficient deterministic PIT for polynomials which can be expressed in the form $C(Q_1, Q_2, \ldots, Q_t)$, where C is a circuit of polynomial degree and $Q_i's$ are either sparse polynomials or product of linear forms, such that the algebraic rank of $\{Q_1, Q_2, \ldots, Q_t\}$[5] is bounded. This approach was extremely powerful as Agrawal et al. [2] show that they can use this approach to recover many of the known PIT results, which otherwise had very different proofs techniques. The PIT results of this paper hold for a variation of the model just described and we describe it in more detail in Section 1.3.2.

Polynomials with low algebraic rank

In addition to potential applications to arithmetic circuit complexity, it seems an interesting mathematical question to understand the structure of a set of algebraically dependent polynomials. In general, our understanding of algebraic dependence is not as clear as our understanding of linear dependence. For instance, we know that if a set of polynomials is linearly dependent, then every polynomial in the set can be written as a linear combination of the polynomials in the basis. However, for higher degree dependencies (linear dependence is dependency of degree 1), we do not know any such clean statement. As a significant core of our proofs, we prove a statement of this flavor in Lemma 1.6.

We now formally define the model of computation studied in this paper, and then state and discuss our results.

1.2 Model of computation

We start with the definition of algebraic dependence. See Section 2 for more details.

▶ **Definition 1.1** (Algebraic independence and algebraic rank)**.** Let $\mathbb{F}$ be any field. A set of polynomials $\mathcal{Q} = \{Q_1, Q_2, \ldots, Q_t\} \subseteq \mathbb{F}[X_1, X_2, \ldots, X_N]$ is said to be algebraically independent over $\mathbb{F}$ if there is no nonzero polynomial $R \in \mathbb{F}[Y_1, Y_2, \ldots, Y_t]$ such that $R(Q_1, Q_2, \ldots, Q_t)$ is identically zero.

A maximal subset of $\mathcal{Q}$ which is algebraically independent is said to be a transcendence basis of $\mathcal{Q}$ and the size of such a set is said to be the algebraic rank of $\mathcal{Q}$.

We are now ready to define the model of computation.

▶ **Definition 1.2.** Let $\mathbb{F}$ be any field. A $\Sigma\Pi^{(k)}\Sigma\Pi$ circuit C in N variables over $\mathbb{F}$ is a representation of an N variate polynomial as

$$C = \sum_{i=1}^{T} Q_{i1} \cdot Q_{i2} \cdot \cdots, Q_{it}$$

[5] See Section 2 for definitions.

such that for each $i \in [T]$, the algebraic rank of the set of polynomials $\{Q_{ij} : j \in [t]\}$ is at most k. Additionally, if for every $i \in [T]$ and $j \in [t]$, the degree of Q_{ij} is at most d, we say that C is a $\Sigma\Pi^{(k)}\Sigma\Pi^{[d]}$ circuit.

We will state all our results for $\Sigma\Pi^{(k)}\Sigma\Pi$ and $\Sigma\Pi^{(k)}\Sigma\Pi^{[d]}$ circuits. However, the results in this paper hold for a more general class of circuits where the product gates at the second level can be replaced by a arbitrary polynomials. This larger class of circuits will be crucially used in our proofs and we define it below. Formally,

▶ **Definition 1.3.** Let $\mathbb{F}$ be any field. A $\Sigma\Gamma^{(k)}\Sigma\Pi$ circuit C in N variables over $\mathbb{F}$ is a representation of an N variate polynomial as

$$C = \sum_{i=1}^{T} \Gamma_i(Q_{i1}, Q_{i2}, \ldots, Q_{it})$$

such that Γ_i is an arbitrary polynomial in t variables, and for each $i \in [T]$, the algebraic rank of the set of polynomials $\{Q_{ij} : j \in [t]\}$ is at most k. Additionally, if for every $i \in [T]$ and $j \in [t]$, the degree of Q_{ij} is at most d, we say that C is a $\Sigma\Gamma^{(k)}\Sigma\Pi^{[d]}$ circuit.

The *size* of a $\Sigma\Pi^{(k)}\Sigma\Pi$ or a $\Sigma\Gamma^{(k)}\Sigma\Pi$ circuit C is defined as the maximum of T and the number of monomials in the set of polynomials $\{Q_{ij} : i \in [T], j \in [t]\}$.

A $\Sigma\Pi^{(k)}\Sigma\Pi$ circuit C for which the polynomials $\{Q_{ij} : i \in [T], j \in [t]\}$ are homogeneous polynomials such that for every $i \in [T]$,

$$\sum_{j \in [t]} \text{Degree}(Q_{ij}) = \text{Degree}(P)$$

(where P is the polynomial being computed) and $k = \text{Degree}(P)$ is precisely the class of homogeneous depth-4 circuits. If we drop the condition of homogeneity, then in general the value of t could be much larger than $\text{Degree}(P)$ as well as the degrees of the Q_{ij} could also be arbitrarily large. Thus the class of $\Sigma\Pi^{(k)}\Sigma\Pi$ circuits with k equalling the degree of the polynomial being computed is potentially a much larger class than that of homogenous depth-4 circuits.

Also note that in the definition of $\Sigma\Pi^{(k)}\Sigma\Pi$ circuits, the bound on the algebraic rank is local for each $i \in [T]$, and in general, the algebraic rank of the entire set $\{Q_{ij} : i \in [T], j \in [t]\}$ can be as large as N.

1.3 Our results

We now state our results and disucss how they relate to other known results.

1.3.1 Lower bounds

As our first result, we show exponential lower bounds on the size of $\Sigma\Pi^{(k)}\Sigma\Pi$ circuits computing an explicit polynomial when the algebraic rank (k) is at most the degree (n) of the polynomial being computed.

▶ **Theorem 1.4.** *Let $\mathbb{F}$ be any field of characteristic zero*[6]*. There exists a family $\{P_n\}$ of polynomials in* VNP*, such that P_n is a polynomial of degree n in $N = n^{O(1)}$ variables with $0, 1$ coefficients, and for any $\Sigma\Pi^{(k)}\Sigma\Pi$ circuit C, if $k \leq n$ and if C computes P_n over $\mathbb{F}$, then*

$$\textit{Size}(C) \geq N^{\Omega(\sqrt{n})}.$$

[6] Sufficiently large characteristic suffices.

▶ Remark. From our proofs it follows that our lower bounds hold for the more general class of $\Sigma\Gamma^{(k)}\Sigma\Pi$ circuits, but for the sake of simplicity, we state our results in terms of $\Sigma\Pi^{(k)}\Sigma\Pi$ circuits. We believe it is likely that the lower bounds also hold for a polynomial in VP and it would be interesting to know if this is indeed true.

▶ Remark. Even though we state Theorem 1.4 for $k \leq n$, the proof goes through as long as k is any polynomial in n and N is chosen to be an appropriately large polynomial in n.

Comparison to known results

As we alluded to in the introduction, $\Sigma\Pi^{(k)}\Sigma\Pi$ circuits for $k \geq n$ subsume the class of homogeneous depth-4 circuits. Therefore, Theorem 1.4 subsumes the lower bounds of [18, 27] for homogeneous depth-4 circuits. Moreover, it also subsumes and generalizes the lower bounds of Agrawal et al. [2] since the [2] lower bounds hold only if the algebraic rank of the entire set of polynomials $\{Q_{ij} : i \in [T], j \in [t]\}$ is bounded, while for Theorem 1.4, we only need upper bounds on the algebraic rank separately for every $i \in [T]$.

1.3.2 Polynomial identity tests

We show that there is a quasipolynomial size hitting set for all polynomials $P \in \Sigma\Pi^{(k)}\Sigma\Pi^{[d]}$ for *bounded* d and k. More formally, we prove the following theorem.

▶ **Theorem 1.5.** *Let $\mathbb{F}$ be any field of characteristic zero*[7]*. Then, for every N, there exists a set $\mathcal{H} \subseteq \mathbb{F}^N$ such that*

$$|\mathcal{H}| \leq \exp(O(\log^{O(1)} N))$$

and for every nonzero N-variate polynomial P over $\mathbb{F}$ which is computable by a $\Sigma\Pi^{(k)}\Sigma\Pi^{[d]}$ circuit with $d, k \leq \log N$ and size $\mathsf{poly}(N)$, there exists an $h \in \mathcal{H}$ such that $P(h) \neq 0$. Moreover, the set $\mathcal{H}$ can be explicitly constructed in time

$$\exp(O(\log^{O(1)} N)).$$

We now mention some remarks about Theorem 1.5.

▶ Remark. It follows from our proof that the hitting set works for the more general class of $\Sigma\Gamma^{(k)}\Sigma\Pi^{[d]}$ circuits with $d, k \leq \log N$, size $\mathrm{poly}(N)$ and formal degree at most $\mathrm{poly}(N)$.

Comparison to known results

The two known results closest to our PIT result are the results of Forbes [8] and the results of Agrawal et al. [2]. Forbes [8] studies PIT for the case where the number of *distinct inputs* to the second level product gates in a depth-4 circuit with bounded bottom fan-in is also bounded (which naturally also bounds the algebraic rank of the inputs), and shows quasipolynomial sized hitting sets for this case. On the other hand, we handle the case where there is no restriction on the number of distinct inputs feeding into the second level product gates, but we need to bound the bottom fan-in as well as the algebraic rank. In this sense, the results in this paper are a generalization of the results in [8].

Agrawal et al. [2] give a construction of polynomial sized hitting sets in the case when the total algebraic rank of the set $\{Q_{ij} : i \in [T], j \in [t]\}$ is bounded, but they can work with

[7] Sufficiently large characteristic suffices.

unbounded d. On the other hand, the size of our hitting set depends exponentially on d, but requires only local algebraic dependencies for every $i \in [T]$. So, these two results are not comparable, although there are similarities in the sense that both of them aim to use the algebraic dependencies in the circuit. In general, summation is a tricky operation with respect to designing PIT algorithms (as opposed to multiplication), so it is not clear if the ideas in [2] can be somehow adapted to prove Theorem 1.5.

1.3.3 From algebraic dependence to functional dependence

Our lower bounds and PIT results crucially use the following lemma, which (informally) shows that over fields of characteristic zero, upto a translation, every polynomial in a set of algebraically dependent polynomials can be written as a *function* of the polynomials in *transcendence basis*[8]. We now state the lemma precisely[9].

▶ **Lemma 1.6** (Algebraic dependence to functional dependence). *Let $\mathbb{F}$ be any field of characteristic zero or sufficiently large characteristic. Let $\mathcal{Q} = \{Q_1, Q_2, \ldots, Q_k, Q_{k+1}\}$ be a set of polynomials in N variables such that the for every $i \in [t]$, the degree of Q_i is equal to d_i and the algebraic rank of $\mathcal{Q}$ equals k. Let $\mathcal{B} = \{Q_1, Q_2, \ldots, Q_k\}$ be a maximal algebraically independent subset of $\mathcal{Q}$. Then, there exists an $\overline{a} = (a_1, a_2, \ldots, a_N)$ in $\mathbb{F}^N$ and a polynomial F_{k+1} in k variables such that*

$$Q_{k+1}(\overline{X} + \overline{a}) = \mathsf{Hom}^{\leq d_{k+1}}\left[F_{k+1}(Q_1(\overline{X} + \overline{a}), Q_2(\overline{X} + \overline{a}), \ldots, Q_k(\overline{X} + \overline{a}))\right] .$$

Even though the lemma seems a very basic statement about the structure of algebraically dependent polynomials, to the best of our knowledge this was not known before. The proof builds upon a result on the structure of roots of multivariate polynomials by Dvir et al. [7]. Observe that for *linear* dependence, the statement analogous to that of Lemma 1.6 is trivially true. We believe that this lemma might be of independent interest (in addition to its applications in this paper).

1.4 Proof overview

Even though the results in this paper seem related to the results in [2] (both exploiting some notion of low algebraic rank), the proof strategy and the way algebraic rank is used are quite different. We now briefly outline our proof strategy.

We first discuss the overview of proof for our lower bound.

Let P_n be the degree n polynomial we want to compute, and let C be a $\Sigma\Pi^{(k)}\Sigma\Pi$ circuit computing it, with $k = n$. Then C can be represented as

$$C = \sum_{i=1}^{T} \prod_{j=1}^{t} Q_{ij} .$$

From definitions, we know that for every $i \in [T]$, the algebraic rank of the set of polynomials $\{Q_{i1}, Q_{i2}, \ldots, Q_{it}\}$ is at most $k(= n)$. We want to show a lower bound on the size of C.

[8] A transcendence basis of a set of polynomials is a maximal subset of the polynomials with the property that its elements are algebraically independent. For more on this see Section 2.

[9] For any polynomial P, we use $\mathsf{Hom}^{\leq i}[P]$ to refer to homogeneous components of P of degree less than or equal to i.

Instead of proving our result directly for $\Sigma\Pi^{(k)}\Sigma\Pi$ circuits, it will be very useful for us to go to the significantly strengthened class of $\Sigma\Gamma^{(k)}\Sigma\Pi$ circuits and prove our result for that class. Thus we think of our circuit C as being expressed as

$$C = \sum_{i=1}^{T} C_i(Q_{i1}, Q_{i2}, \ldots, Q_{it})$$

where the C_i can be arbitrary polynomial functions of the inputs feeding into them. Note that we define the size of a $\Sigma\Gamma^{(k)}\Sigma\Pi$ circuit to be the maximum of the top fan-in T, and the maximum of the number of monomials in any of the polynomials Q_{ij} feeding into the circuit. Thus we completely disregard the complexities of the various polynomial function gates at the second level. If we are able to prove a lower bound for this notion of size, then if the original circuit is actually a $\Sigma\Pi^{(k)}\Sigma\Pi$ circuit then it will also be as good a lower bound for the usual notion of size.

Our lower bound has two key steps. In the first step we prove the result in the special case where $t \leq n^2$. In the second step we show how to "almost" reduce to the case of $t \leq n^2$.

Step 1: $t \leq n^2$

In the representation of C as a $\Sigma\Gamma^{(k)}\Sigma\Pi$ circuit, the value of t is at most n^2. Lower bounds for this case turn out to be similar to lower bounds for homogeneous depth-4 circuits. In this case we borrow ideas from prior works [14, 18, 27] and show that the *dimension of projected shifted partial derivatives of* C is not too large. Most importantly, we can use the chain rule for partial derivatives to obtain good bounds for this complexity measure, independent of the complexity of the various C_i.

Recall however that in our final result, t can be actually much larger than n^2. Indeed the circuit C can be very far from being homogeneous, and for general depth-4 circuits, we do not know good upper bounds on the complexity of shifted partial derivatives or projected shifted partial derivatives. Also, in general, it is not clear if these measures are really small for general depth-4 circuits[10]. It is here that the low algebraic rank of $\{Q_{i1}, Q_{i2}, \ldots, Q_{it}\}$ proves to be useful, and that brings us to the crux of our argument.

Step 2 : Reducing to the case where $t \leq n^2$

A key component of our proof, which is formalized in Lemma 3.5 shows that over any field of characteristic zero (or sufficiently large characteristic), upto a translation, every polynomial in a set of algebraically dependent polynomials can be written as a function of the homogeneous components of the polynomials in the transcendence basis.

More formally, there exists an $\overline{a} \in \mathbb{F}^N$ such that $C(\overline{X} + \overline{a})$ can be expressed as

$$C(\overline{X} + \overline{a}) = \sum_{i=1}^{T} C'_i(\mathsf{Hom}[Q_{i1}(\overline{X} + \overline{a})], \mathsf{Hom}[Q_{i2}(\overline{X} + \overline{a})], \ldots, \mathsf{Hom}[Q_{ik}(\overline{X} + \overline{a})]) \quad (1)$$

where for a degree d polynomial F, $\mathsf{Hom}[F]$ denotes the $d+1$-tuple of homogeneous components of F.

The crucial gain in the above transformation is that the arity of each of the polynomials C'_i is $(d+1) \times k$ and not t (where d is an upper bound on the degrees of the Q_{ij}). Now

[10] Indeed, as [26] shows, even homogeneous depth-4 circuits can have very large shifted partial derivative complexity.

by assumption $k \leq n$, and moreover WLOG we can assume $d \leq n$ since homogeneous components of Q_{ij} of degree larger than n can be dropped since they do not contribute to the computation of a degree n polynomial. Thus we have essentially reduced to the case where $t \leq n^2$.

One loss by this transformation is that the polynomials $\{C_i'\}$ might be much more complex and with much higher degrees than the original polynomials $\{C_i\}$. However this will not affect the computation of our complexity measure. Another loss is that we have to deal with the translated polynomial $C(\overline{X} + \overline{a})$. This introduces some subtleties into our computation as it could be that $Q_{ij}(\overline{X})$ is a sparse polynomial but $Q_{ij}(\overline{X} + \overline{a})$ is far from being sparse. Neither of these issues is very difficult to deal with, and we are able to get strong bounds for the projected shifted partial derivative based measure for such circuits. The proof of Lemma 3.5 essentially follows from Lemma 1.6, and seems critical for the proof.

The proof of Lemma 1.6 crucially uses a result of Dvir, Shpilka and Yehudayoff [7] which shows that upto some minor technical conditions (which are not very hard to satisfy), factors of a polynomial $f \in \mathbb{F}[X_1, X_2, \ldots, X_N, Y]$ of the form $Y - p(X_1, X_2, \ldots, X_N)$ where $p \in \mathbb{F}[X_1, X_2, \ldots, X_N]$ can be expressed as polynomials in the coefficients when viewing f as an element of $\mathbb{F}[X_1, X_2, \ldots, X_N][Y]$. This is relevant since a set t of polynomials are algebraically dependent implies that there is a non-zero t-variate polynomial which vanishes when composed with this tuple. We use this *vanishing* to prove the lemma.

The PIT results follows a similar initial setup and use of Lemma 1.6. We then use a result of [8] to show that the polynomial computed by C has a monomial of small support, which is then detected using the standard idea of using Shpilka-Volkovich generators [40].

1.5 Organization of the paper

The rest of the paper is organized as follows: In Section 2, we state some preliminary definitions and results that are used elsewhere in the paper. In Section 3, we describe our use of low algebraic rank and prove Lemma 3.5. We prove Theorem 1.4 in Section 4 and Theorem 1.5 in Section 5.

2 Preliminaries

In this section we set up some notations and definitions for the rest of the paper.

Notations

1. For an integer i, we denote the set $\{1, 2, \ldots, i\}$ by $[i]$.
2. By $\overline{X}$, we mean the set of variables $\{X_1, X_2, \ldots, X_N\}$.
3. For a polynomial P and a positive integer i, we represent by $\mathsf{Hom}^i[P]$, the homogeneous component of P of degree equal to i. By $\mathsf{Hom}^{\leq i}[P]$ and $\mathsf{Hom}^{\geq i}[P]$, we represent the component of P of degree at most i and at least i respectively. We define $\mathsf{Hom}[P]$ as the ordered tuple of homogeneous components of P, i.e $\mathsf{Hom}[P] = \left(\mathsf{Hom}^d[P], \mathsf{Hom}^{d-1}[P], \ldots, \mathsf{Hom}^0[P]\right)$, where d is the degree of P. If for some i, there are no non-zero monomials of degree equal to i in P, then $\mathsf{Hom}^i[P] = 0$.
4. The support of a monomial α is the set of variables which appear with a non-zero exponent in α. We denote the size of the support of α by $\mathrm{Supp}(\alpha)$.
5. We say that a function $f(N)$ is quasipolynomial in N if there exists a positive absolute constant c, such that for all N sufficiently large, $f(N) < \exp(\log^c N)$.
6. In this paper, unless otherwise stated, $\mathbb{F}$ is a field of characteristic zero.

7. Given a polynomial P and a valid monomial ordering Π, the leading monomial of P is the monomial with a nonzero coefficient in P which is maximal according to Π. Similarly, the trailing monomial in P is the monomial which is minimal among all monomials in P according to Π.

Algebraic independence

We now formally define the notions of algebraic independence, algebraic rank and transcendence basis which would be widely used in this paper.

▶ **Definition 2.1** (Algebraic independence and algebraic rank)**.** Let $\mathbb{F}$ be any field. A set of polynomials $\mathcal{Q} = \{Q_1, Q_2, \ldots, Q_t\} \subseteq \mathbb{F}[X_1, X_2, \ldots, X_N]$ is said to be algebraically independent over $\mathbb{F}$ if there is no nonzero polynomial $R \in \mathbb{F}[Y_1, Y_2, \ldots, Y_t]$ such that $R(Q_1, Q_2, \ldots, Q_t)$ is identically zero.

A maximal subset of $\mathcal{Q}$ which is algebraically independent is said to be a transcendence basis of $\mathcal{Q}$ and the size of such a set is said to be the algebraic rank of $\mathcal{Q}$.

Apriori, it is not even clear that algebraic rank of a set of polynomials is well defined. But it is known that algebraic independence satisfies the matroid property [29], and therefore is well defined.

For a tuple $\mathcal{Q} = (Q_1, Q_2, \ldots, Q_t)$ of algebraically dependent polynomials, we know that there is a nonzero t variate polynomial R (called a $\mathcal{Q}$-annihilating polynomial) such that $R(Q_1, Q_2, \ldots, Q_t)$ is identically zero. A natural question is to ask, what kind of bounds on the degree of R can we show, in terms of the degrees of Q_i. The following lemma of Kayal [17] shows an upper bound on the degree of annihilating polynomials of a set of degree d polynomials. The bound is useful to us in our proof.

▶ **Lemma 2.2** (Kayal [17])**.** *Let $\mathbb{F}$ be a field and let $\mathcal{Q} = (Q_1, Q_2, \ldots, Q_t)$ be a set of polynomials of degree d in N variables over the field $\mathbb{F}$ having algebraic rank k. Then there exists a $\mathcal{Q}$-annihilating polynomial of degree at most $(k+1) \cdot d^k$.*

Complexity of homogeneous components

We will use the following simple lemma (whose proof is fairly standard using interpolation), and can be found in [28] for instance. We sketch the proof here for completeness.

▶ **Lemma 2.3.** *Let $\mathbb{F}$ be a field of characteristic zero, and let $P \in \mathbb{F}[X_1, X_2, \ldots, X_N]$ be a polynomial of degree at most d, in N variables, such that P can be represented as*

$$P = C(Q_1, Q_2, \ldots, Q_t)$$

where for every $j \in [t]$, Q_j is a polynomial in N variables, and C is an arbitrary polynomial in t variables. Then, there exist polynomials $\{Q'_{ij} : i \in [d+1], j \in [t]\}$, and for every ℓ such that $0 \leq \ell \leq d$, there exist polynomials $C'_{\ell,1}, C'_{\ell,2}, \ldots, C'_{\ell,d+1}$ satisfying

$$\mathsf{Hom}^{\ell}[P] = \sum_{i=1}^{(d+1)} C'_{\ell,i}(Q'_{i1}, Q'_{i2}, \ldots, Q'_{it}) \,.$$

Moreover,

- *if each of the polynomials in the set $\{Q_j : j \in [t]\}$ is of degree at most Δ, then every polynomial in the set $\{Q'_{ij} : i \in [d+1], j \in [t]\}$ is also of degree at most Δ.*

- *if the algebraic rank of the set of polynomials $\{Q_j : j \in [t]\}$ is at most k, then for every $i \in [d+1]$, the algebraic rank of polynomials in the set $\{Q'_{ij} : j \in [t]\}$ is also at most k.*

Proof. The key idea is to start from $P \in \mathbb{F}[\overline{X}]$ and obtain a new polynomial $P' \in \mathbb{F}[\overline{X}][Z]$ such that for every ℓ such that $0 \leq \ell \leq d$, the coefficient of Z^ℓ in P' equals $\mathsf{Hom}^\ell[P]$. Here, Z is a new variable. Such a P' is obtained by replacing every occurence of the variable X_j (for each $j \in [N]$) in P by $Z \cdot X_j$. It is not hard to verify that such a P' has the stated property. We now view P' as a univariate polynomial in Z with the coefficients coming from $\mathbb{F}(\overline{X})$. Notice that the degree of P' in Z is at most d. So, to recover the coefficients of a univariate polynomial of degree at most d, we can evaluate P' at $d+1$ distinct values of Z over $\mathbb{F}(\overline{X})$ and take an $\mathbb{F}(\overline{X})$ linear combination. In fact, if the field $\mathbb{F}$ is large enough, we can assume that all these distinct values of Z lie in the base field $\mathbb{F}$ and we only take an $\mathbb{F}$ linear combination. The properties in the 'moreover' part of the lemma immediately follow from this construction, and we skip the details. ◀

Roots of polynomials

We will crucially use the following result of Dvir, Shpilka, Yehudayoff [7].

▶ **Lemma 2.4** (Dvir, Shpilka, Yehudayoff [7]). *For a field $\mathbb{F}$, let $P \in \mathbb{F}[X_1, X_2, \ldots, X_N, Y]$ be a non-zero polynomial of degree at most k in Y. Let $f \in \mathbb{F}[X_1, X_2, \ldots, X_N]$ be a polynomial such that $P(X_1, X_2, \ldots, X_N, f) = 0$ and $\frac{\partial P}{\partial Y}(0, 0, \ldots, 0, f(0, 0, \ldots, 0)) \neq 0$. Let*

$$P = \sum_{i=0}^{k} C_i(X_1, X_2, \ldots, X_N) \cdot Y^i \,.$$

Then, for every $t \geq 0$, there exists a polynomial $R_t \in \mathbb{F}[Z_1, Z_2, \ldots, Z_{k+1}]$ of degree at most t such that

$$\mathsf{Hom}^{\leq t}[f(X_1, X_2, \ldots, X_N)] = \mathsf{Hom}^{\leq t}[R_t(C_0, C_1, \ldots, C_k)] \,.$$

We also use the following standard result about zeroes of polynomials.

▶ **Lemma 2.5** (Schwartz, Zippel, DeMillo, Lipton). *Let P be a non-zero polynomial of degree d in N variables over a field $\mathbb{F}$. Let S be an arbitrary subset of $\mathbb{F}$, and let $x_1, x_2, \ldots, x_N$ be random elements from S chosen independently and uniformly at random. Then*

$$Pr[P(x_1, x_2, \ldots, x_N) = 0] \leq \frac{d}{|S|} \,.$$

The following corollary easily follows from the lemma above.

▶ **Corollary 2.6.** *Let $P_1, P_2, \ldots, P_t$ be non-zero polynomials of degree d in N variables over a field $\mathbb{F}$. Let S be an arbitrary subset of $\mathbb{F}$ of size at least $2td$, and let $x_1, x_2, \ldots, x_N$ be random elements from S chosen independently and uniformly at random. Then*

$$Pr[\forall i \in [t], P_i(x_1, x_2, \ldots, x_N) \neq 0] \geq \frac{1}{2} \,.$$

Approximations

We will use the following lemma of Saptharishi [32] for numerical approximations in our calculations.

▶ **Lemma 2.7** (Saptharishi [32]). *Let n and ℓ be parameters such that $\ell = \frac{n}{2}(1-\epsilon)$ for some $\epsilon = o(1)$. For any a, b such that $a, b = O(\sqrt{n})$,*

$$\binom{n-a}{\ell-b} = \binom{n}{\ell} \cdot 2^{-a} \cdot (1+\epsilon)^{a-2b} \cdot \exp(O(b \cdot \epsilon^2))$$

3 Utilizing low algebraic rank

Let $\mathcal{Q} = \{Q_1, Q_2, \ldots, Q_t\}$ be a set of polynomials in N variables and degree at most d such that the algebraic rank of $\mathcal{Q}$ equals k. Without loss of generality, let us assume that $\mathcal{B} = \{Q_1, Q_2, \ldots, Q_k\}$ are an algebraically independent subset of $\mathcal{C}$ of maximal size. We now show that, in some sense, this implies that all the polynomials in $\mathcal{Q}$ can be represented as functions of polynomials in the set $\mathcal{B}$. We make this notion formal in the following lemma.

▶ **Lemma 3.1** (Algebraic dependence to functional dependence). *Let $\mathbb{F}$ be any field of characteristic zero or sufficiently large. Let $\mathcal{Q} = \{Q_1, Q_2, \ldots, Q_t\}$ be a set of polynomials in N variables such that the for every $i \in [t]$, the degree of Q_i is equal to d_i and the algebraic rank of $\mathcal{Q}$ equals k. Let $\mathcal{B} = \{Q_1, Q_2, \ldots, Q_k\}$ be a maximal algebraically independent subset of $\mathcal{Q}$. Then, there exists an $\overline{a} = (a_1, a_2, \ldots, a_N)$ in $\mathbb{F}^N$ and polynomials $F_{k+1}, F_{k+2}, \ldots, F_t$ in k variables such that $\forall i \in \{k+1, k+2, \ldots, t\}$*

$$Q_i(\overline{X} + \overline{a}) = \mathsf{Hom}^{\leq d_i}\left[F_i(Q_1(\overline{X} + \overline{a}), Q_2(\overline{X} + \overline{a}), \ldots, Q_k(\overline{X} + \overline{a}))\right] .$$

Proof. Let d be defined as $max_i\{d_i\}$. Let us consider any i such that $i \in \{k+1, k+2, \ldots, t\}$. From the statement of the lemma, it follows that the set of polynomials in the set $\mathcal{B} \cup \{Q_i\}$ are algebraically dependent. Therefore, there exists a nonzero polynomial A_i in $k+1$ variables such that $A_i(Q_1, Q_2, \ldots, Q_k, Q_i) \equiv 0$. Without loss of generality, we choose such a polynomial with the smallest total degree. From the upper bound on the degree of the annihilating polynomial from Lemma 2.2, we can assume that the degree of A_i is at most $(k+1)d^k$. Consider the polynomial $A'_i(\overline{X}, Y)$ defined by

$$A'_i(\overline{X}, Y) = A_i(Q_1(\overline{X}), Q_2(\overline{X}), \ldots, Q_k(\overline{X}), Y) .$$

We have the following observation about properties of A'_i.

▶ **Observation 3.2.** *A'_i satisfies the following properties:*

- *A'_i is not identically zero*
- *The Y degree of A'_i is at least one.*
- *$Q_i(\overline{X})$ is a root of the polynomial A'_i, when viewing it as a polynomial in the Y variable with coefficients coming from $\mathbb{F}(\overline{X})$.*

Proof. We prove the items in sequence:

- If A'_i is identically zero, then it follows that $Q_1, Q_2, \ldots, Q_k$ are algebraically dependent, which is a contradiction.
- If $A'_i(\overline{X}, Y)$ does not depend on the variable Y, then by definition, it follows that $A_i(Q_1, Q_2, \ldots, Q_k, Y)$ does not depend on Y. Hence, $A_i(Q_1, Q_2, \ldots, Q_k, Q_i)$ does not depend on Q_i but is identically zero. This contradicts the algebraic independence of $Q_1, Q_2, \ldots, Q_k$.
- This item follows from the fact that the polynomial obtained by substituting Y by Q_i in A'_i equals $A_i(Q_1, Q_2, \ldots, Q_k, Q_i)$, which is identically zero. ◀

Our aim now is to invoke Lemma 2.4 for the polynomial A'_i, but first, we need to verify that the conditions in the hypothesis of Lemma 2.4 are satisfied. Let the polynomial A''_i be defined as the first order derivative of A'_i with respect to Y. Formally,

$$A''_i = \frac{\partial A'_i}{\partial Y} .$$

We proceed with the following claim, the proof of which we defer to the end.

▶ **Claim 3.3.** *The polynomial A_i'' is not an identically zero polynomial and $A_i''|_{Y=Q_i}$ is not identically zero.*

For the ease of notation, we define

$$L_i(\overline{X}) = A_i''|_{Y=Q_i}\,.$$

Observe that L_i is a polynomial in the variables $\overline{X}$ which is not identically zero and is of degree at most $(k+1)d^{k+1}$. Let H be a subset of $\mathbb{F}$ of size $2t(k+1)d^{k+1}$. Then, for a uniformly random point $\overline{a}_i$ picked from H^N, the probability that L_i vanishes at $\overline{a}_i$ is at most $1/2t$. We call the set of all points $\overline{a}_i \in H^N$ where L_i vanishes as bad. Then, with a probability at least $1 - 1/2t$, a uniformly random element of H^N is not bad. Let $\overline{a}_i \in \mathbb{F}^N$ be a 'not bad' element . We can replace X_j by $X_j + \overline{a}_{i_j}$ and then for the resulting polynomial $L_i(\overline{X} + \overline{a}_i)$, the point $(0, 0, \ldots, 0)$ is not bad.

We are now ready to apply Lemma 2.4. Let

$$A_i'(\overline{X}, Y) = \sum_{j=0}^{(k+1)d^k} C_j(\overline{X}) \cdot Y^j\,.$$

Here, for every j, $C_j(\overline{X}) = C_j'\left(Q_1(\overline{X}), Q_2(\overline{X}), \ldots, Q_k(\overline{X})\right)$ is a polynomial in the $\overline{X}$ variables and is the coefficient of Y^j in $A_i'(\overline{X}, Y)$ when viewed as an element of $\mathbb{F}[\overline{X}][Y]$. From the discussion above, we know that the following are true.

1. The polynomial $A_i'(\overline{X} + \overline{a}_i, Q_i(\overline{X} + \overline{a}_i))$ is identically zero.
2. The first derivative of $A_i'(\overline{X} + \overline{a}_i, Y)$ with respect to Y does not vanish at $(0, 0, \ldots, 0, Q_i(0, 0, \ldots, 0))$.

Therefore, by Lemma 2.4, it follows that there is a polynomial G_i such that

$$Q_i(\overline{X} + \overline{a}_i) = \mathsf{Hom}^{\leq d_i}\left[G_i(C_0(\overline{X} + \overline{a}_i), C_1(\overline{X} + \overline{a}_i), \ldots, C_{(k+1)d^k}(\overline{X} + \overline{a}_i))\right]\,.$$

We also know that for every $j \in \{0, 1, \ldots, (k+1)d^k\}$, $C_j(\overline{X} + \overline{a}_i)$ is a polynomial in the polynomials $Q_1(\overline{X} + \overline{a}_i), Q_2(\overline{X} + \overline{a}_i), \ldots, Q_k(\overline{X} + \overline{a}_i)$. In other words,

$$Q_i(\overline{X} + \overline{a}_i) = \mathsf{Hom}^{\leq d_i}\left[F_i(Q_1(\overline{X} + \overline{a}_i), Q_2(\overline{X} + \overline{a}_i), \ldots, Q_k(\overline{X} + \overline{a}_i))\right]$$

for a polynomial F_i.

In order to prove the lemma for all values of $i \in \{k+1, k+2, \ldots, t\}$, we observe that we can pick a single value of the translation $\overline{a}$, which works for every $i \in \{k+1, k+2, \ldots, t\}$. Such an $\overline{a}$ exists because the probability that a uniformly random $p \in H^N$ is bad for some i is at most $t \cdot 1/2t = 1/2$ and the translation corresponding to any such element $\overline{a}$ in H^N which is not bad for every i will work. The statement of the lemma then immediately follows. ◀

We now prove Claim 3.3.

Proof of Claim 3.3. We observed from the second item in Observation 3.2 that the degree of Y in A_i' is at least 1. Hence, A_i'' is not identically zero. If $A_i''|_{Y=Q_i}$ is identically zero, then it follows that $\{Q_1, Q_2, \ldots, Q_k, Q_i\}$ have an annihilating polynomial of degree smaller than the degree of A_i, which is a contradiction to the choice of A_i, as a minimum degree annihilating polynomial. ◀

Lemma 3.1 lets us express all polynomials in a set of polynomials as a function of the polynomials in the transcendence basis. However, the functional form obtained is slightly cumbersome for us to use in our applications. We now derive the following corollary, which is easier to use in our applications.

▶ **Corollary 3.4.** *Let $\mathbb{F}$ be any field of characteristic zero or sufficiently large. Let $\mathcal{Q} = \{Q_1, Q_2, \ldots, Q_t\}$ be a set of polynomials in N variables such that the for every $i \in [t]$, the degree of Q_i is equal to $d_i < d$ and the algebraic rank of $\mathcal{Q}$ equals k. Let $\mathcal{B} = \{Q_1, Q_2, \ldots, Q_k\}$ be a maximal algebraically independent subset of $\mathcal{Q}$. Then, there exists an $\overline{a} = (a_1, a_2, \ldots, a_N)$ in $\mathbb{F}^N$ and polynomials $F_{k+1}, F_{k+2}, \ldots, F_t$ in at most $k(d+1)$ variables such that $\forall i \in \{k+1, k+2, \ldots, t\}$*

$$Q_i(\overline{X} + \overline{a}) = \left[F_i(\mathsf{Hom}[Q_1(\overline{X} + \overline{a})], \mathsf{Hom}[Q_2(\overline{X} + \overline{a})], \ldots, \mathsf{Hom}[Q_k(\overline{X} + \overline{a})])\right] .$$

Proof. Let i be such that $i \in \{k+1, k+2, \ldots, t\}$. From Lemma 3.1, we know that there exists an $\overline{a} \in \mathbb{F}^N$ and a polynomial W_i such that

$$Q_i(\overline{X} + \overline{a}) = \mathsf{Hom}^{\leq d_i}\left[W_i(Q_1(\overline{X} + \overline{a}), Q_2(\overline{X} + \overline{a}), \ldots, Q_k(\overline{X} + \overline{a}))\right]$$

We will now show that $\mathsf{Hom}^{\leq d_i}\left[W_i(Q_1(\overline{X} + \overline{a}), Q_2(\overline{X} + \overline{a}), \ldots, Q_k(\overline{X} + \overline{a}))\right]$ is actually a polynomial in the homogeneous components of the various $Q_j(\overline{X} + \overline{a})$ by the following procedure, which is essentially univariate polynomial interpolation.

- Let $R(\overline{X}) = W_i(Q_1(\overline{X} + \overline{a}), Q_2(\overline{X} + \overline{a}), \ldots, Q_k(\overline{X} + \overline{a}))$. We replace every variable X_j in R by $Z \cdot X_j$ for a new variable Z. We view the resulting polynomial R' as an element of $\mathbb{F}(\overline{X})[Z]$, i.e a univariate polynomial in Z with coefficients coming from the field of rational functions in the $\overline{X}$ variables.
- Now, observe that for any ℓ, the homogeneous component of degree ℓ of R is precisely the coefficient of Z^ℓ in R'. Hence, we can evaluate R' for sufficiently many distinct values of Z in $\mathbb{F}(\overline{X})$, and then take an $\mathbb{F}(\overline{X})$ linear combination of these evaluations to express the homogeneous components. Moreover, since $\mathbb{F}$ is an infinite field, without loss of generality, we can pick the values of Z to be scalars in $\mathbb{F}$, and in this case, we will just be taking an $\mathbb{F}$ linear combination.

The catch here is that after replacing X_j by $Z \cdot X_j$ and substituting different values of $Z \in \mathbb{F}$, the polynomials $Q_{i'}(\overline{X} + \overline{a})$ could possibly lead to distinct polynomials. In general, this is bad, since our goal is to show that every polynomial in a set of algebraically dependent polynomials in a function of *few* polynomials. However, the following observation comes to our rescue. Let P be any polynomial in $\mathbb{F}[\overline{X}]$ of degree Δ and let P' be the polynomial obtained from P by replacing X_j by $Z \cdot X_j$. Then,

$$P'(\overline{X} + \overline{a}) = \sum_{\ell=0}^{\Delta} Z^\ell . \mathsf{Hom}^\ell[P(\overline{X})]$$

In particular, the set of polynomials obtained from P' for different values of Z are all in the linear span of homogeneous components of P.

Therefore, any homogeneous component of R can be expressed as a function of the set of polynomials $\cup_{i=1}^{k} \mathsf{Hom}\left[Q_i(\overline{X} + \overline{a})\right]$. This completes the proof of the corollary. ◀

We now prove the following lemma, which will be directly useful in the our applications to polynomial identity testing and lower bounds in the following sections.

▶ **Lemma 3.5.** *Let $\mathbb{F}$ be any field of characteristic zero or sufficiently large. Let $P \in \mathbb{F}[\overline{X}]$ be a polynomial in N variables, of degree equal to n, such that P can be represented as*

$$P = \sum_{i=1}^{T} F_i(Q_{i1}, Q_{i2}, \ldots, Q_{it})$$

and such that the following are true

- *For each $i \in [T]$, F_i is a polynomial in t variables.*
- *For each $i \in [T]$ and $j \in [t]$, Q_{ij} is a polynomial in N variables of degree at most d.*
- *For each $i \in [T]$, the algebraic rank of the set of polynomials $\{Q_{ij} : j \in [t]\}$ is at most k and $\mathcal{B}_i = \{Q_{i1}, Q_{i2}, \ldots, Q_{ik}\}$ is a maximal algebraically independent subset of $\{Q_{ij} : j \in [t]\}$.*

Then, there exists an $\overline{a} \in \mathbb{F}^N$ and polynomials F'_i in at most $k(d+1)$ variables such that

$$P(\overline{X} + \overline{a}) = \sum_{i=1}^{T} F'_i(\mathsf{Hom}[Q_{i1}(\overline{X} + \overline{a})], \mathsf{Hom}[Q_{i2}(\overline{X} + \overline{a})], \ldots, \mathsf{Hom}[Q_{ik}(\overline{X} + \overline{a})]) \,. \quad (2)$$

Proof. The proof would essentially follow from the application of Corollary 3.4 to each of the summands on the right hand side. The only catch is that the transalations $\overline{a}$ could be different for each one of them. Since we are working over infinite fields, without loss of generality, we can assume that there is a good translation $\overline{a}$ which works for all the summands. ◀

4 Application to lower bounds

In this section , we prove Theorem 1.4. But, first we discuss the definitions of the complexity measure used in the proof, the notion of random restrictions and the family of hard polynomials that we work with.

4.1 Projected shifted partial derivatives

The complexity measure that we use to prove the lower bounds in this paper is the notion of *projected shifted partial derivatives* of a polynomial introduced in [18] and subsequently used in a number of following papers [27, 22, 28].

For a polynomial P and a monomial γ, $\frac{\partial P}{\partial \gamma}$ is the partial derivative of P with respect to γ and for a set of monomials $\mathcal{M}$, $\partial_{\mathcal{M}}(P)$ is the set of partial derivatives of P with respect to monomials in $\mathcal{M}$. The space of $(\mathcal{M}, m)$-projected shifted partial derivatives of a polynomial P is defined below.

▶ **Definition 4.1** ($(\mathcal{M}, m)$-projected shifted partial derivatives)**.** For an N variate polynomial $P \in \mathbb{F}[X_1, X_2, \ldots, X_N]$, set of monomials $\mathcal{M}$ and a positive integer $m \geq 0$, the space of $(\mathcal{M}, m)$-projected shifted partial derivatives of P is defined as

$$\langle \partial_{\mathcal{M}}(P) \rangle_m \stackrel{def}{=} \mathbb{F}\text{-}span\{\mathsf{Mult}\left[\prod_{i \in S} X_i \cdot g\right] : g \in \partial_{\mathcal{M}}(P), S \subseteq [N], |S| = m\} \,. \quad (3)$$

Here, $\mathsf{Mult}[P]$ of a polynomial P is the projection of P on the multilinear monomials in its support. We use the dimension of projected shifted partial derivative space of P with respect to some set of monomials $\mathcal{M}$ and a parameter m as a measure of the complexity of a polynomial. Formally,

$$\Phi_{\mathcal{M},m}(P) = \mathsf{Dim}(\langle \partial_{\mathcal{M}}(P) \rangle_m) \,.$$

From the definitions, it is straight forward to see that the measure is subadditive.

▶ **Lemma 4.2** (Sub-additivity)**.** *Let P and Q be any two multivariate polynomials in $\mathbb{F}[X_1, X_2, \ldots, X_N]$. Let $\mathcal{M}$ be any set of monomials and m be any positive integer. Then, for all scalars α and β*

$$\Phi_{\mathcal{M},m}(\alpha \cdot P + \beta \cdot Q) \leq \Phi_{\mathcal{M},m}(P) + \Phi_{\mathcal{M},m}(Q) \,.$$

In the proof of Theorem 1.4, we need to upper bound the dimension of the span of projected shifted partial derivatives of the homogeneous component of a fixed degree of polynomials. The following lemma comes to our rescue there.

▶ **Lemma 4.3.** *Let P be a polynomial of degree at most d. Then for every $0 \leq i \leq d$, and for every choice of parameters m, r and a set $\mathcal{M}$ of monomials of degree equal to r, the following inequality is true*

$$\phi_{\mathcal{M},m}(P) \geq \phi_{\mathcal{M},m}(\mathsf{Hom}^i[P]) .$$

Proof. Since $\mathcal{M}$ is a subset of monomials of degree equal to r, all the partials derivatives are shifted by monomials of degree equal to m and the operation $\mathsf{Mult}[]$ either sets a monomial to zero or leaves it unchanged, it follows that the span of projected shifted partial derivatives of $\mathsf{Hom}^i[P]$ coincides with the span of the homogeneous components of degree $(i-r)m$ in the space of span of projected shifted partial derivatives of P itself. The lemma then follows from the fact that dimension of a linear space of polynomials is at least as large as the dimension of the space obtained by restricting all polynomials to some fixed homogeneous component. ◀

In the next lemma, we prove an upper bound on the polynomials which are obtained by a composition of low arity polynomials with polynomials of small support. Gupta et al. [14] first proved such a bound for homogeneous depth-4 circuit with bounded bottom fan-in.

▶ **Lemma 4.4.** *Let s be a parameter and $Q_1, Q_2, \ldots, Q_t$ be polynomials in $\mathbb{F}[\overline{X}]$ such that for every $i \in [t]$, the support of every monomial in Q_i is of size at most s. Then, for every polynomial F in t variables, every choice of parameters r, m such that $m + rs \leq N/2$, and every set $\mathcal{M}$ of monomials of degree equal to r,*

$$\Phi_{\mathcal{M},m}(F(Q_1, Q_2, \ldots, Q_t)) \leq \binom{t+r}{r} \cdot \binom{N}{m+rs} .$$

Proof. By the chain rule for partial derivatives, every derivative of order r of $F(Q_1, Q_2, \ldots, Q_t)$ can be written as a linear combination of products of the form

$$\left(\frac{\partial F(Y_1, Y_2, \ldots, Y_t)}{\partial \beta_0} |_{Y_i = Q_i} \right) \cdot \prod_{1 \leq j \leq r} \frac{\partial P_j}{\partial \beta_j}$$

where

1. β_0 is a monomial in variables $Y_1, Y_2, \ldots, Y_t$ of degree at most r
2. for every $1 \leq j \leq r$, the polynomial P_j is an element of $\{Q_1, Q_2, \ldots, Q_t\}$, and
3. for every $1 \leq j \leq r$, β_j is a monomial in variables $X_1, X_2, \ldots, X_N$

Since every monomial in each Q_i is of support at most s, every monomial in each of the products $\prod_{1 \leq j \leq r} \frac{\partial P_j}{\partial \beta_j}$ is of support at most rs. Therefore, for shifts of degree m, the projected shifted partial derivatives of $F(Q_1, Q_2, \ldots, Q_t)$ (with respect to monomials in $\mathcal{M}$ which are of degree r) are in the linear span of polynomials of the form

$$\mathsf{Mult}\left[\left(\frac{\partial F(Y_1, Y_2, \ldots, Y_t)}{\partial \beta_0} |_{Y_i = Q_i} \right) \cdot \alpha \right]$$

where α is a multilinear monomial of degree at most $m + rs$. Therefore, the dimension of this space is upper bounded by the number of possible choices of β_0 and α. Hence

$$\Phi_{\mathcal{M},m}(F(Q_1, Q_2, \ldots, Q_t)) \leq \binom{t+r}{r} \cdot \binom{N}{m+rs} .$$ ◀

4.2 Target polynomials for the lower bound

In this section, we define the family of polynomials for which we show our lower bounds. The family is a variant of the Nisan-Wigderson polynomials which were introduced by Kayal et al. in [23], and subsequently used in many other results [27, 22, 28]. We start with the following definition.

▶ **Definition 4.5** (Nisan-Wigderson polynomial families)**.** Let n, q, e be arbitrary parameters with q being a power of a prime, and $n, e \leq q$. We identify the set $[q]$ with the field $\mathbb{F}_q$ of q elements. Observe that since $n \leq q$, we have that $[n] \subseteq \mathbb{F}_q$. The Nisan-Wigderson polynomial with parameters n, q, e, denoted by $\mathsf{NW}_{n,q,e}$ is defined as

$$\mathsf{NW}_{n,q,e}(\overline{X}) = \sum_{\substack{p(t) \in \mathbb{F}_q[t] \\ \mathsf{Deg}(p) < e}} X_{1,p(1)} \dots X_{n,p(n)} \,.$$

The number of variables in $\mathsf{NW}_{n,q,e}$ as defined above is $N = q \cdot n$. The lower bounds in this paper will be proved for the polynomial $\mathsf{NW} \circ \mathsf{Lin}$ which is a variant of the polynomial $\mathsf{NW}_{n,q,e}$ defined as follows.

▶ **Definition 4.6** (Hard polynomials for the lower bound)**.** Let $\delta \in (0, 1)$ be an arbitrary constant, and let $p = N^{-\delta}$. Let

$$\gamma = \frac{N}{p} \,.$$

The polynomial $\mathsf{NW} \circ \mathsf{Lin}_{\mathsf{q,n,e,p}}$ is defined as

$$\mathsf{NW} \circ \mathsf{Lin}_{\mathsf{q,n,e,p}} = \mathsf{NW}_{q,n,e} \left(\sum_{i=1}^{\gamma} X_{1,1,i}, \sum_{i=1}^{\gamma} X_{1,2,i}, \dots, \sum_{i=1}^{\gamma} X_{n,q,i} \right) .$$

For brevity, we will denote $\mathsf{NW} \circ \mathsf{Lin}_{\mathsf{q,n,e,p}}$ by $\mathsf{NW} \circ \mathsf{Lin}$ for the rest of the discussion.

The advantage of using this trick[11] of composing with linear forms is that it becomes cleaner to show that the polynomial $\mathsf{NW} \circ \mathsf{Lin}$ is robust under random restrictions where every variable is kept alive with a probability p. Since δ is an absolute constant, the number of variables in $\mathsf{NW} \circ \mathsf{Lin}$ is at most $N^{O(1)}$. We now formally define our notion of random restrictions.

Let $\mathcal{V}$ be the set of variables in the polynomial $\mathsf{NW} \circ \mathsf{Lin}$. We now define a distribution $\mathcal{D}_p$ over the subsets of $\mathcal{V}$.

The distribution $\mathcal{D}_p$: Each variable in $\mathcal{V}$ is independently kept alive with a probability $p = N^{-\delta}$.

The random restriction procedure samples a $V \leftarrow \mathcal{D}$ and then keeps only the variables in V alive. The remaining variables are set to 0. We denote the restriction of the polynomial obtained by such a restriction as $\mathsf{NW} \circ \mathsf{Lin}|_V$. Observe that a random restriction also results in a distribution over the restrictions of a circuit computing the polynomial $\mathsf{NW} \circ \mathsf{Lin}$. We denote by $C|_V$ the restriction of a circuit C obtained by setting every input gate in C which is labeled by a variable outside V to 0.

We now show that with a high probability over restrictions sampled according to $\mathcal{D}_p$, the projected shifted partial derivative complexity of $\mathsf{NW} \circ \mathsf{Lin}$ remains high. We need the following lower bound on the dimension of projected shifted partial derivatives of $\mathsf{NW}_{n,q,e}$.

[11] This idea came up during discussions with Ramprasad Saptharishi.

▶ **Lemma 4.7** ([27, 25]). *For every n and $r = O(\sqrt{n})$ there exists parameters q, e, ϵ such that $q = \Omega(n^2)$, $N = qn$ and $\epsilon = \Theta\left(\frac{\log n}{\sqrt{n}}\right)$ with*

$$\begin{aligned} q^r &\geq (1+\epsilon)^{2(n-r)} \\ q^{e-r} &= \left(\frac{2}{1+\epsilon}\right)^{n-r} \cdot \mathit{poly}(q). \end{aligned}$$

For any $\{n, q, e, r, \epsilon\}$ satisfying the above constraints, for $m = \frac{N}{2}(1-\epsilon)$, over any field $\mathbb{F}$, we have

$$\Phi(\mathsf{NW}_{n,q,e}) \geq \binom{N}{m+n-r} \cdot \exp(-O(\log^2 n)).$$

We will instantiate the lemma above with the following choice of parameters:

- $\epsilon = \frac{4 \log n}{\sqrt{n}}$
- $r = \sqrt{n}$
- $q = n^{10}$
- Besides, we will set the parameter $s = \frac{\sqrt{n}}{100}$

It is straight forward to check that for the above choice of parameters, there is a choice of e such that

$$\begin{aligned} q^r &\geq (1+\epsilon)^{2(n-r)} \\ q^{e-r} &= \left(\frac{2}{1+\epsilon}\right)^{n-r} \cdot \mathsf{poly}(q). \end{aligned}$$

Therefore, for $m = \frac{N}{2}(1-\epsilon)$, over any field $\mathbb{F}$, we have

$$\Phi(\mathsf{NW}_{n,q,e}) \geq \binom{N}{m+n-r} \cdot \exp(-O(\log^2 n)).$$

We are now ready to prove our main lemma for this section.

▶ **Lemma 4.8.** *With a probability at least $1 - o(1)$ over $V \leftarrow \mathcal{D}_p$, there exists a subset of variables $V' \subseteq V$ such that $|V'| = N$ and*

$$\Phi(\mathsf{NW} \circ \mathsf{Lin}|_{V'}) \geq \binom{N}{m+n-r} \cdot \exp(-O(\log^2 n)).$$

Proof. To prove the lemma, we first show that with a high probability over the random restrictions, the polynomial $P|_V$ has the polynomial $\mathsf{NW}_{n,q,e}$ as a $0,1$ projection. Combining this with Lemma 4.7 would complete the proof. We now fill in the details.

Let $i \in [N]$. Then, the probability that all the variables in the set $A_{i,j} = \{X_{i,j,\ell} : \ell \in [\gamma]\}$ are set to zero by the random restrictions is equal to $(1-p)^\gamma \leq \exp(-\Theta(N))$. Therefore, the probability that there exists an $i \in [n], j \in [q]$ such that all the variables in the set $A_{i,j}$ are set to zero by the random restrictions, is at most $N \cdot \exp(-\Theta(N)) = o(1)$. We now argue that if this event does not happen (which is the case with probability at least $1 - o(1)$), then the dimension of the projected shifted partial derivatives is large.

For every i, j, let $A'_{i,j}$ be the subset of $A_{i,j}$ which has not been set to zero. We know that for every i, j, $A'_{i,j}$ is non-empty. Now, for every i, j, we set all the elements of $A'_{i,j}$ to zero except one. Observe that the polynomial obtained from $\mathsf{NW} \circ \mathsf{Lin}$ after this restriction is exactly the polynomial $NW_{n,q,e}$ upto a relabeling of variables. Now, from Lemma 4.7, our claim follows. ◀

4.3 Proof of Theorem 1.4

To show our lower bound, we show that under random restrictions from the distribution $\mathcal{D}_p$, the dimension of the linear span of projected shifted partial derivatives of any $\Sigma\Pi^{(n)}\Sigma\Pi$ circuit C is small with a high probability if the size of the C is *not too large*. Comparing this with the lower bound on the dimension of projected shifted partials of the polynomial $\mathsf{NW} \circ \mathsf{Lin}$ under random restrictions from Lemma 4.8, the lower bound follows. We now proceed along this outline and prove the following lemma.

▶ **Lemma 4.9** (Upper bound on complexity of circuits)**.** *Let m, r, s be parameters such that $m+rs \leq N/2$. Let $\mathcal{M}$ be any set of multilinear monomials of degree r. Let C be an arithmetic circuit computing a homogeneous polynomial of degree n such that*

$$C = \sum_{i=1}^{T} C_i(Q_{i1}, Q_{i2}, \ldots, Q_{it})$$

where

- *For each $i \in [T]$, C_i is a polynomial in t variables.*
- *For each $i \in [T]$ and $j \in [t]$, Q_{ij} is a homogeneous polynomial in N variables.*
- *For each $i \in [T]$, the algebraic rank of the set of polynomials $\{Q_{ij} : j \in [t]\}$ is at most k.*

For each $i \in [T]$ and $j \in [t]$, let S_{ij} be the set of monomials with nonzero coefficients in Q_{ij}. If

$$\left| \bigcup_{i \in [T], j \in [t]} S_{ij} \right| \leq N^{\frac{\delta s}{2}}$$

then, with a probability at least $1 - o(1)$ over $V \leftarrow \mathcal{D}_p$[12] for all subsets V' of V of size at most N

$$\Phi(C|_{V'}) \leq T \binom{k(n+1)+r}{r} \binom{N}{m+rs}.$$

Proof. We prove the lemma by first using random restrictions to simplify the circuit into one with bounded bottom support, and then utilizing the tools tools developed in Section 3 and Section 4.1 to conclude that the dimension of the space of projected shifted partial derivatives of the resulting circuit is small.

Step 1: Random restrictions

From the definition of random restrictions, every variable is kept alive independently with a probability $p = N^{-\delta}$. So, the probability that a monomial of support at least s survives the restrictions is at most $N^{-\delta s}$. Therefore, by linearity of expectations, the expected number of monomials of support at least s in $\bigcup_{i \in [T], j \in [t]} S_{ij}$ which survive the random restrictions is at most

$$\left| \bigcup_{i \in [T], j \in [t]} S_{ij} \right| \cdot N^{-\delta s} \leq N^{-\frac{\delta s}{2}}.$$

[12] This is the distribution defined in Section 4.2, where every variable is kept alive with a probability $N^{-\delta}$ for a constant $\delta \in (0, 1)$.

So, by Markov's inequality, the probability that at least one monomial of support at least s in $\bigcup_{i\in[T],j\in[t]} S_{ij}$ survives the random restrictions is $o(1)$. Let V' be any subset of the surviving set of variables of size N. For the rest of the proof, we assume that all the variables outside the set V' are set to zero. Restrictions which kill all monomials in $\bigcup_{i\in[T],j\in[t]} S_{ij}$ are said to be good.

Step 2: Using low algebraic rank

In this step, we assume that we are given a good restriction C' of the circuit C. Let

$$C' = \sum_{i=1}^{T} C'_i(Q'_{i1}, Q'_{i2}, \ldots, Q'_{it})$$

where for every $i \in [T], j \in [t]$, all monomials of Q'_{ij} have support at most s. Observe that random restrictions cannot increase the algebraic rank of a set of polynomials. Therefore, for every $i \in [T]$, the algebraic rank of the set of polynomials $\{Q'_{ij} : j \in [t]\}$ is at most k. For ease of notation, let us assume that the algebraic rank is equal to k. Without loss of generality, let the set $\mathcal{B}_i = \{Q'_{i1}, Q'_{i2}, \ldots, Q'_{ik}\}$ be the set guaranteed by Lemma 3.5. We know that there exists an $\overline{a} \in \mathbb{F}^N$ and polynomials $\{F_i : i \in [T]\}$ such that

$$C'(\overline{X} + \overline{a}) = \sum_{i=1}^{T} F'_i(\mathsf{Hom}\left[Q'_{i1}(\overline{X} + \overline{a})\right], \mathsf{Hom}\left[Q'_{i2}(\overline{X} + \overline{a})\right], \ldots, \mathsf{Hom}\left[Q'_{ik}(\overline{X} + \overline{a})\right]).$$

Moreover, since $C(\overline{X})$ (and hence $C'(\overline{X})$) is a homogeneous polynomial of degree n, the following is true:

$$C'(\overline{X}) = \mathsf{Hom}^n\left[\sum_{i=1}^{T} F'_i(\mathsf{Hom}\left[Q'_{i1}(\overline{X} + \overline{a})\right], \mathsf{Hom}\left[Q'_{i2}(\overline{X} + \overline{a})\right], \ldots, \mathsf{Hom}\left[Q'_{ik}(\overline{X} + \overline{a})\right])\right]. \tag{4}$$

An important observation here is that for the rest of the argument, we can assume that the degree of every polynomial $Q'_{ij}(\overline{X} + \overline{a})$ is at most n. If not, we can simply replace any such high degree $Q'_{ij}(\overline{X} + \overline{a})$ by $\mathsf{Hom}^{\leq n}\left[Q'_{ij}(\overline{X} + \overline{a})\right]$. We claim that the equality 4 continues to hold. This is because the higher degree monomials of Q_{ij} do not participate in the computation of the lower degree monomials. The only monomials which could potentially change by this substitution are the ones with degree strictly larger than n.

Step 3: Upper bound on $\Phi_{\mathcal{M},m}(C'(\overline{X}))$

Let R be defined the polynomial

$$R = \left[\sum_{i=1}^{T} F'_i(\mathsf{Hom}\left[Q'_{i1}(\overline{X} + \overline{a})\right], \mathsf{Hom}\left[Q'_{i2}(\overline{X} + \overline{a})\right], \ldots, \mathsf{Hom}\left[Q'_{ik}(\overline{X} + \overline{a})\right])\right].$$

From Lemma 4.4 and from Lemma 4.2, it is easy to see that

$$\Phi_{\mathcal{M},m}(R) \leq T\binom{k(n+1)+r}{r}\binom{N}{m+rs}.$$

From Lemma 4.3, it follows that

$$\Phi_{\mathcal{M},m}(C'(\overline{X})) \leq \Phi_{\mathcal{M},m}(R) \leq T\binom{k(n+1)+r}{r}\binom{N}{m+rs}.$$

Observe that steps 2 and 3 of the proof are always successful if the restriction in step 1 is good, which happens with a probability at least $1 - o(1)$. So, the lemma follows. ◀

We now complete the proof of Theorem 1.4.

Proof of Theorem 1.4. If the size of the circuit C is at least $N^{\frac{\delta}{2}\sqrt{n}}$, then we are done. Else, the size of C is at most $N^{\frac{\delta}{2}\sqrt{n}}$. This implies that the total number of monomials in all the polynomials Q_{ij} together is at most $N^{\frac{\delta}{2}\sqrt{n}}$. From Lemma 4.9 and Lemma 4.8, it follows that with a nonzero probability, there exists a subset V' of variables of size N such that both the following inequalities are true:

$$\Phi_{\mathcal{M},m}(C|_{V'}) \leq T\binom{k(n+1)+r}{r}\binom{N}{m+rs} \tag{5}$$

and

$$\Phi_{\mathcal{M},m}(\mathsf{NW} \circ \mathsf{Lin}|_{V'}) \geq \binom{N}{m+n-r} \cdot \exp(-\log^2 n)\,.$$

Since C computes $\mathsf{NW} \circ \mathsf{Lin}$, it must be the case that

$$T \geq \frac{\binom{N}{m+n-r} \cdot \exp(-\log^2 n)}{\binom{k(n+1)+r}{r}\binom{N}{m+rs}}\,.$$

Plugging in the value of the parameters, and approximating using Lemma 2.7, we immediately get

$$\binom{N}{m+n-r} = \binom{N}{m} \cdot (1+\epsilon)^{2(n-r)} \cdot exp(O((n-r)\cdot\epsilon^2))$$

and

$$\binom{N}{m+rs} = \binom{N}{m} \cdot (1+\epsilon)^{2rs} \cdot exp(O(rs\cdot\epsilon^2))\,.$$

Moreover, $\binom{k(n+1)+r}{r} \leq (nk)^r \leq \exp(2\sqrt{n}\cdot\log n)$. Taking the ratio and substituting the values of the parameters, we get

$$T \geq \exp\left(\Omega(\sqrt{n}\log N)\right). \qquad \blacktriangleleft$$

5 Application to polynomial identity testing

In this section we give an application of the ideas developed in Section 3 to the question of polynomial identity testing and prove Theorem 1.5. We start by formally defining the notion of a hitting set.

Hitting set

Let $\mathcal{S}$ be a set of polynomials in N variables over a field $\mathbb{F}$. Then, a set $\mathcal{H} \subseteq \mathbb{F}^N$ is said to be a *hitting set* for the class $\mathcal{S}$, if for every polynomial $P \in \mathcal{S}$ such that P is not identically zero, there exists a $p \in \mathcal{H}$ such that $P(p) \neq 0$.

For our PIT result, we show that any nonzero polynomial P in the circuit class we consider, has a monomial of low support. A hitting set can then be constructed by the standard techniques using the Shpilka-Volkovich generator [39].

▶ **Lemma 5.1** (Shpilka-Volkovich generator [40]). *Let $\mathbb{F}$ be a field of characteristic zero. For every ℓ, d, N, there exists a set $\mathcal{H} \subseteq \mathbb{F}^N$ of size at most $(O(Nd))^\ell$ such that for every nonzero polynomial P of degree at most d in N variables which contains a monomial of support at most ℓ, there exists an $h \in \mathcal{H}$ such that $P(h) \neq 0$. Moreover, the set $\mathcal{H}$ can be constructed in time poly$(N, d, \ell) \cdot (O(Nd))^\ell$.*

The following lemma is our main technical claim.

▶ **Lemma 5.2.** *Let $\mathbb{F}$ be a field of characteristic zero. Let P be a homogeneous polynomial of degree Δ in N variables such that P can be represented as*

$$P = \sum_{i=1}^{T} C_i(Q_{i1}, Q_{i2}, \ldots, Q_{it})$$

such that the following are true

- *For each $i \in [T]$, C_i is a polynomial in t variables.*
- *For each $i \in [T]$ and $j \in [t]$, Q_{ij} is a polynomial of degree at most d in N variables.*
- *For each $i \in [T]$, the algebraic rank of the set of polynomials $\{Q_{ij} : j \in [t]\}$ is at most k.*

Then, the trailing monomial of P has support at most

$$2e^3 d \cdot (\ln(T(\Delta+1)) + (d+1)k \ln(2(d+1)k) + 1).$$

Here, e is the Euler's constant.

In order to prove Lemma 5.2, we follow the outline of proving *robust* lower bounds for arithmetic circuits, described and used by Forbes [8]. This essentially amounts to showing that the trailing monomial of P has small support. We use the following result of Forbes [8] in a blackbox manner which greatly simplifies our proof.

▶ **Lemma 5.3** (Forbes [8]). *Let $R(\overline{X})$ be a polynomial in $\mathbb{F}[\overline{X}]$ such that*

$$R(\overline{X}) = \sum_{i=1}^{T} F_i(Q_{i1}, Q_{i2}, \ldots, Q_{it})$$

and for each $i \in [T]$ and $j \in [j]$, the degree of Q_{ij} is at most d. Let α be the trailing monomial of R. Then, the support of α is at most $2e^3 d(\ln T + t \ln 2t + 1)$, where e is the Euler's constant.

We now proceed to prove Lemma 5.2.

Proof of Lemma 5.2. Recall that our goal is to show that the polynomial P, which can be represented as

$$P = \sum_{i=1}^{T} C_i(Q_{i1}, Q_{i2}, \ldots, Q_{it})$$

has a trailing monomial of small support.

For every $i \in [T]$, let $\mathcal{Q}_i = \{Q_{i1}, Q_{i2}, \ldots, Q_{it}\}$ and let $\mathcal{Q}_i$ be of algebraic rank k_i. Without loss of generality, let us assume the sets $\mathcal{B}_i = \{Q_{i1}, Q_{i2}, \ldots, Q_{ik_i}\}$ are the sets guaranteed by Lemma 3.5. This implies that there exist polynomials $F_1, F_2, \ldots, F_T$ and $\overline{a} \in \mathbb{F}^N$ such that

$$P(\overline{X} + \overline{a}) = \left[\sum_{i=1}^{T} F_i(\mathsf{Hom}[Q_{i1}(\overline{X} + \overline{a})], \mathsf{Hom}[Q_{i2}(\overline{X} + \overline{a})], \ldots, \mathsf{Hom}[Q_{ik_i}(\overline{X} + \overline{a})]) \right].$$

Since each $k_i \leq k$, for the ease of notation, we assume that each $k_i = k$. Observe that if P is a homogeneous polynomial of degree $\mathrm{Deg}(P) \leq \Delta$, then,

$$\mathsf{Hom}^{\mathrm{Deg}(P)}[P(\overline{X} + \overline{a})] \equiv P(\overline{X}) .$$

So, from Lemma 2.3, it follows that there exist $k(d+1)$ variate polynomials $F'_1, F'_2, \ldots, F'_{T(\Delta+1)}$ and a set of polynomials $\{Q'_{ij} : i \in [T(\Delta+1)], j \in [k]\}$ such that

$$P(\overline{X}) = \mathsf{Hom}^{\mathrm{Deg}(P)} \left[\sum_{i=1}^{T(\Delta+1)} F'_i(\mathsf{Hom}[Q'_{i1}(\overline{X} + \overline{a})], \mathsf{Hom}[Q'_{i2}(\overline{X} + \overline{a})], \ldots, \mathsf{Hom}[Q'_{ik}(\overline{X} + \overline{a})]) \right] .$$

Moreover, every polynomial in the set $\{Q'_{ij} : i \in [T(\Delta+1)], j \in [k]\}$ has degree at most d. Now, Lemma 5.3 implies that the trailing monomial α of $P(\overline{X})$ has support at most

$$2e^3 d \cdot (\ln(T(\Delta+1)) + (d+1)k \ln(2(d+1)k) + 1) . \quad \blacktriangleleft$$

We are now ready to complete the proof of Theorem 1.5.

Proof of Theorem 1.5. From Definition 1.2, it follows there could be non-homogeneous polynomials $P \in \mathcal{C}$. So, we cannot directly use Lemma 5.2 to say something about them, since the proof relies on homogeneity. But, this is not a problem, since a polynomial is identically zero if and only if all its homogeneous components are identically zero. Moreover, by applying Lemma 2.3 to every summand feeding into the top sum gate of the circuit, we get that every homogeneous component of P[13] can also be computed by a circuit similar in structure to that of P at the cost of a blow up by a factor $\Delta + 1$ in the top fan-in. We can then apply Lemma 5.2 to each of these homogeneous components to conclude that if P is not identically zero, then it contains a monomial of support at most

$$2e^3 d \cdot (\ln\left(T(\Delta+1)^2\right) + (d+1)k \ln(2(d+1)k) + 1) .$$

Theorem 1.5 immediately follows by detecting the low support monomial using Lemma 5.2 and Lemma 5.1. $\blacktriangleleft$

6 Open questions

We end with some open questions:

- One question is to prove the lower bounds in the paper for a polynomial in VP. We believe this is true, but it seems that we need a strengthening of the bounds in [27]. In particular, it needs to be shown that the lower bound for IMM (Iterated matrix multiplication) continues to hold when a depth-4 circuit is not homogeneous but the formal degree is at most the square of the degree of the polynomial itself.
- An intriguing consequence of the proofs in the paper is that the characteristic of the underlying field needs to be high or zero. In particular, we do not know if Lemma 3.1 is true over fields of low characteristic. In general, we seem to have a slightly better understanding of algebraic dependence over fields of large characteristic or characteristic zero. For instance, as far as we know the results of Agrawal et al. [2] are not known to extend to fields of low characteristic since the Jacobian condition for algebraic independence fails there. We wonder if our proof techniques also suffer from a similar technical obstacle.

[13] Only the top fan-in increases by a factor of $\Delta + 1$, all other parameters in Definition 1.2 remain the same.

- It would be interesting to see if there are other applications of Lemma 1.6 to questions in complexity theory. The Jacobian characterization of algebraic independence has several very interesting applications [2, 6].

Acknowledgements. Many thanks to Ramprasad Saptharishi for answering numerous questions regarding the results and techniques in [2]. We are also thankful to Michael Forbes for sharing a draft of his paper [8] with us. We would also like to thank the anonymous reviewers at CCC-2016 for their comments which helped improve the presentation of the paper.

References

1 M. Agrawal and V. Vinay. Arithmetic circuits: A chasm at depth four. In *FOCS*, 2008.

2 Manindra Agrawal, Chandan Saha, Ramprasad Saptharishi, and Nitin Saxena. Jacobian hits circuits: hitting-sets, lower bounds for depth-d occur-k formulas & depth-3 transcendence degree-k circuits. In *Proceedings of the 44th ACM symposium on Theory of computing*, pages 599–614, 2012.

3 Malte Beecken, Johannes Mittmann, and Nitin Saxena. Algebraic independence and blackbox identity testing. In *Automata, Languages and Programming - 38th International Colloquium, ICALP 2011, Zurich, Switzerland, July 4-8, 2011, Proceedings, Part II*, pages 137–148, 2011.

4 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.

5 Z. Dvir and A. Shpilka. Locally decodable codes with 2 queries and polynomial identity testing for depth 3 circuits. *SIAM Journal on Computing*, 36(5):1404–1434, 2006.

6 Zeev Dvir, Ariel Gabizon, and Avi Wigderson. Extractors and rank extractors for polynomial sources. *computational complexity*, 18(1):1–58, 2009. URL: `http://dx.doi.org/10.1007/s00037-009-0258-4`, `doi:10.1007/s00037-009-0258-4`.

7 Zeev Dvir, Amir Shpilka, and Amir Yehudayoff. Hardness-randomness tradeoffs for bounded depth arithmetic circuits. *SIAM J. Comput.*, 39(4):1279–1293, 2009. URL: `http://dx.doi.org/10.1137/080735850`, `doi:10.1137/080735850`.

8 Michael Forbes. Deterministic divisibility testing via shifted partial derivatives. In *FOCS*, 2015.

9 Michael A. Forbes and Amir Shpilka. Quasipolynomial-time identity testing of noncommutative and read-once oblivious algebraic branching programs. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS*, pages 243–252, 2013.

10 Michael A. Forbes and Amir Shpilka. Quasipolynomial-time identity testing of noncommutative and read-once oblivious algebraic branching programs. *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, 0:243–252, 2013.

11 H. Fournier, N. Limaye, G. Malod, and S. Srinivasan. Lower bounds for depth 4 formulas computing iterated matrix multiplication. In *STOC*, 2014.

12 D. Grigoriev and 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.

13 Dima Grigoriev and Marek Karpinski. An exponential lower bound for depth 3 arithmetic circuits. In *Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing (STOC)*, pages 577–582, 1998.

14 A. Gupta, P. Kamath, N. Kayal, and R. Saptharishi. Approaching the chasm at depth four. In *CCC*, 2013.

15 Z. Karnin, P. Mukhopadhyay, A. Shpilka, and I. Volkovich. Deterministic identity testing of depth 4 multilinear circuits with bounded top fan-in. In *Proceedings of the 42nd Annual STOC*, pages 649–658, 2010.

16 Z. Karnin and A. Shpilka. Black box polynomial identity testing of generalized depth-3 arithmetic circuits with bounded top fan-in. In *In proceedings of 23rd Annual CCC*, pages 280–291, 2008.

17 N. Kayal. The complexity of the annihilating polynomial. In *Computational Complexity, 2009. CCC '09. 24th Annual IEEE Conference on*, pages 184–193, July 2009. doi:10.1109/CCC.2009.37.

18 N. Kayal, N. Limaye, C. Saha, and S. Srinivasan. An exponential lower bound for homogeneous depth four arithmetic formulas. In *FOCS*, 2014.

19 N. Kayal and S. Saraf. Blackbox polynomial identity testing for depth 3 circuits. In *Proceedings of the 50th Annual FOCS*, pages 198–207, 2009.

20 N. Kayal and N. Saxena. Polynomial identity testing for depth 3 circuits. *Computational Complexity*, 16(2):115–138, 2007.

21 Neeraj Kayal. An exponential lower bound for the sum of powers of bounded degree polynomials. *ECCC*, 19:81, 2012.

22 Neeraj Kayal and Chandan Saha. Lower bounds for depth three arithmetic circuits with small bottom fanin. *Electronic Colloquium on Computational Complexity (ECCC)*, 21:89, 2014. URL: http://eccc.hpi-web.de/report/2014/089.

23 Neeraj Kayal, Chandan Saha, and Ramprasad Saptharishi. A super-polynomial lower bound for regular arithmetic formulas. In *STOC*, 2014.

24 P. Koiran. Arithmetic circuits : The chasm at depth four gets wider. *TCS*, 2012.

25 Mrinal Kumar and Ramprasad Saptharishi. An exponential lower bound for homogeneous depth-5 circuits over finite fields. *ECCC*, 2015.

26 Mrinal Kumar and Shubhangi Saraf. The limits of depth reduction for arithmetic formulas: It's all about the top fan-in. In *STOC*, 2014.

27 Mrinal Kumar and Shubhangi Saraf. On the power of homogeneous depth 4 arithmetic circuits. *FOCS*, 2014.

28 Mrinal Kumar and Shubhangi Saraf. Sums of products of polynomials in few variables : lower bounds and polynomial identity testing. *Electronic Colloquium on Computational Complexity (ECCC)*, 2015.

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

30 Ran Raz. Elusive functions and lower bounds for arithmetic circuits. *Theory of Computing*, 6(1):135–177, 2010.

31 R. Saptharishi. Recent progress on arithmetic circuit lower bounds. *Bulletin of the EATCS*, 2014.

32 R. Saptharishi. A survey of lower bounds in arithmetic circuit complexity. *Manuscript*, 2014.

33 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.

34 N. Saxena and C. Seshadhri. From sylvester-gallai configurations to rank bounds: Improved black-box identity test for deph-3 circuits. In *Proceedings of the 51st Annual FOCS*, pages 21–30, 2010.

35 N. Saxena and C. Seshadhri. Blackbox identity testing for bounded top-fanin depth-3 circuits: The field doesn't matter. *SIAM Journal on Computing*, 41(5):1285–1298, 2012.

36 A. Shpilka and A. Wigderson. Depth-3 arithmetic circuits over fields of characteristic zero. *Computational Complexity*, 10(1):1–27, 2001.

37 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, March 2010. doi:10.1561/0400000039.

38 Amir Shpilka. Affine projections of symmetric polynomials. In *Proceedings of the 16th Annual Conference on Computational Complexity*, CCC '01, pages 160–, Washington, DC, USA, 2001. IEEE Computer Society.

39 Amir Shpilka and Ilya Volkovich. Improved polynomial identity testing for read-once formulas. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, 12th International Workshop, APPROX 2009, and 13th International Workshop, RANDOM 2009, Berkeley, CA, USA, August 21-23, 2009. Proceedings*, pages 700–713, 2009.

40 Amir Shpilka and Ilya Volkovich. Improved polynomial identity testing of read-once formulas. In *Approximation, Randomization and Combinatorial Optimization.Algorithms and Techniques, volume 5687 of LNCS*, pages 700–713, 2009. URL: http://www.cs.technion.ac.il/~shpilka/publications/PROF.pdf.

41 Sébastien Tavenas. Improved bounds for reduction to depth 4 and depth 3. In *MFCS*, pages 813–824, 2013.

42 L. G. Valiant. Completeness classes in algebra. In *STOC*, 1979.

Sums of Products of Polynomials in Few Variables: Lower Bounds and Polynomial Identity Testing

Mrinal Kumar[*1] **and Shubhangi Saraf**[†2]

1 **Department of Computer Science, Rutgers University, New Brunswick, USA**
mrinal.kumar@rutgers.edu

2 **Department of Computer Science and Department of Mathematics, Rutgers University, New Brunswick, USA**
shubhangi.saraf@gmail.com

Abstract

We study the complexity of representing polynomials as a sum of products of polynomials in few variables. More precisely, we study representations of the form $P = \sum_{i=1}^{T} \prod_{j=1}^{d} Q_{ij}$ such that each Q_{ij} is an arbitrary polynomial that depends on at most s variables.

We prove the following results.

- Over fields of characteristic zero, for every constant μ such that $0 \leq \mu < 1$, we give an explicit family of polynomials $\{P_N\}$, where P_N is of degree n in $N = n^{O(1)}$ variables, such that any representation of the above type for P_N with $s = N^{\mu}$ requires $Td \geq n^{\Omega(\sqrt{n})}$. This strengthens a recent result of Kayal and Saha [17] which showed similar lower bounds for the model of sums of products of linear forms in few variables. It is known that any asymptotic improvement in the exponent of the lower bounds (even for $s = \sqrt{n}$) would separate VP and VNP [17].
- We obtain a deterministic subexponential time blackbox polynomial identity testing (PIT) algorithm for circuits computed by the above model when T and the individual degree of each variable in P are at most $\log^{O(1)} N$ and $s \leq N^{\mu}$ for any constant $\mu < 1/2$. We get quasipolynomial running time when $s < \log^{O(1)} N$. The PIT algorithm is obtained by combining our lower bounds with the hardness-randomness tradeoffs developed in [6, 14]. To the best of our knowledge, this is the first nontrivial PIT algorithm for this model (even for the case $s = 2$), and the first nontrivial PIT algorithm obtained from lower bounds for small depth circuits.[1]

1998 ACM Subject Classification F.2.1 Numerical Algorithms and Problems, I.1.1 Expressions and Their Representation

Keywords and phrases arithmetic circuits, lower bounds, polynomial identity testing, hardness vs randomness

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.35

1 Introduction

Arithmetic circuits are the most natural model of computation for a wide variety of algebraic problems such as matrix multiplication, computing fast fourier transforms etc. The problem

* Research supported in part by NSF grants CCF-1350572 and by Simons Graduate Fellowship.
† Research supported by NSF grant CCF-1350572.

[1] In a recent independent work, Forbes [7] does blackbox identity testing for another subclass of depth four circuits using shifted partial derivative based methods. To the best of our understanding, the results in these two papers are incomparable even though both rely on similar techniques.

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 35; pp. 35:1–35:29

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

of proving lower bounds for arithmetic circuits is one of the most fundamental and interesting problems in complexity theory. Proving superpolynomial lower bounds for general arithmetic circuits would resolve the VP versus VNP conjecture [34], the algebraic analog of the P vs NP conjecture. This is one of the holy grails of complexity theory and has received a lot of attention, since it is a more structured and potentially easier question to understand and analyse than the P vs NP problem .

The intimately related problem of polynomial identity testing (PIT) is the problem of testing if a polynomial, given as an arithmetic circuit is identically zero. In the setting where the algorithm cannot look inside the circuit, but only has access to evaluations of the circuit, the problem is referred to as blackbox PIT. There is a very simple randomized algorithm for this problem - simply evaluate the polynomial at a random point from a large enough domain. With very high probability, a nonzero polynomial will have a nonzero evaluation [30, 36]. It is a very important and fundamental question to derandomize the above algorithm. In a seminal work, Kabanets and Impagliazzo [14] showed that the problem of proving lower bounds for arithmetic circuits and the problem of derandomizing identity testing are essentially equivalent[2]!

These two problems have occupied a central position in complexity theory and despite much attention, our understanding of general arithmetic circuits is still very limited. Thus there has been a great deal of effort in understanding the complexity of restricted classes of arithmetic circuits in an attempt to obtain a better understanding of the general problem. Low depth arithmetic circuits in particular are one such well studied class.

Lower bounds for homogeneous low depth arithmetic circuits

The last few years have seen a tremendous amount of exciting progress on the problems of "depth reduction" of general arithmetic circuits to low depth arithmetic circuits, and of proving lower bounds for low depth arithmetic circuits. Using depth reduction techniques [35, 1, 20, 33] it was shown that $N^{\omega(\sqrt{n})}$ lower bounds (for polynomials in N variables and of degree n) for just homogeneous depth 4 arithmetic circuits of bottom fan-in $\sqrt{n}$ would suffice to separate VP from VNP and imply superpolynomial lower bounds for general arithmetic circuits. At the same time there was a very exciting line of works proving $N^{\Omega(\sqrt{n})}$ lower bounds for the same model of arithmetic circuits (and in fact for even the more general class of homogeneous depth 4 arithmetic circuits with no restriction on bottom fan-in) [11, 10, 19, 21, 15, 22].

Lower bounds for non-homogeneous low depth arithmetic circuits

Despite all this remarkable progress, and some very strong lower bounds for homogeneous low depth arithmetic circuits, in the nonhomogenous world much less is understood. Only mild lower bounds are known when we drop the condition of homogeneity, even for very simple classes of low depth arithmetic circuits. For depth 3 circuits over fields of characteristic 0, only quadratic lower bounds known [31, 32], and there has been no progress on this question in more than a decade now.

In a beautiful depth reduction result over fields of characteristic 0, Gupta et al [13] showed that $N^{\omega(\sqrt{n})}$ lower bounds (for polynomials in N variables and of degree n) for the class of non-homogeneous *depth 3* circuits would already separate VP from VNP. It was recently observed by Kayal and Saha [17] [3] that in fact it suffices to prove such lower bounds for depth 3 circuits with bottom fan-in $\sqrt{n}$.

---

[2] They non-trivially transferred such known tradeoffs from the boolean world to the arithmetic world[25].
[3] They attribute the observation to Ramprasad Saptharishi.

Till recently (in particular till the work of [17]), the best known lower bounds for depth 3 circuits even with bottom fan-in 2 were still just quadratic. In a very nice recent result, Kayal and Saha [17] showed an exponential lower bound for depth 3 circuits over fields of characteristic 0, whose bottom fan-in is at most N^{μ}, where N is the number of variables and $0 \leq \mu < 1$ is an arbitrary constant. More precisely, they prove the following.

▶ **Theorem 1.1** (Kayal-Saha [17]). *Let $\mathbb{F}$ be a field of characteristic zero. Then, for every constant $0 \leq \mu < 1$ there is a family $\{P_N\}$ of degree n polynomials in $N = n^{O_\mu(1)}$ variables over $\mathbb{F}$ in* VNP *such that any depth three circuit of bottom fan-in at most N^{μ} computing P_N has top fan-in at least $N^{\Omega_\mu(\sqrt{n})}$.*

Our Model

In this work, we consider the model of sums of products of polynomials in few variables. More formally, we consider representations of polynomials P (degree n in $N = n^{O(1)}$ variables) in the form

$$P = \sum_{i=1}^{T} \prod_{j=1}^{d} Q_{ij} \tag{1}$$

where each Q_{ij} is an arbitrary polynomial (of arbitrarily high degree) in at most s variables. We call this the model of $\Sigma\Pi\left(\Sigma\Pi\right)^{[s]}$ circuits.

Observe that the model is more general than that considered in [17]. The model in [17] corresponds to sums of products of *linear forms* in few variables. In our case, the Q_{ij} no longer have to be linear forms, but can be general polynomials of arbitrarily high degree. Prior to this work, even for the case when $s = 2$, there were no nontrivial lower bounds known for this model.

$\Sigma\Pi\left(\Sigma\Pi\right)^{[s]}$ circuits for $s \geq 2$ can also be seen as a generalization of the model of sums of products of univariate polynomials (which corresponds to $\Sigma\Pi\left(\Sigma\Pi\right)^{[s]}$ circuits with $s = 1$), which has been very well studied in the arithmetic circuit complexity literature. Lower bounds for $\Sigma\Pi\left(\Sigma\Pi\right)^{[1]}$ circuits follow from works of Nisan [24] and Saxena [29]. Over the last few years, there have been some very nice results giving quasipolynomial time blackbox identity testers for $\Sigma\Pi\left(\Sigma\Pi\right)^{[1]}$ circuits [8, 9, 3]. $\Sigma\Pi\left(\Sigma\Pi\right)^{[s]}$ circuits can also be seen as a generalization of the widely studied model of diagonal circuits, since polynomials computable by diagonal circuits can be represented as a $\Sigma\Pi\left(\Sigma\Pi\right)^{[1]}$ circuit without much blow up in the size of the representation [29].

Although $\Sigma\Pi\left(\Sigma\Pi\right)^{[1]}$ circuits seem fairly well understood from the point of view of lower bounds and derandomization of polynomial identity testing, if one considers the model of sums of products of bivariate polynomials ($\Sigma\Pi\left(\Sigma\Pi\right)^{[2]}$ circuits), then our understanding changes completely. Although only seemingly a mild generalization of $\Sigma\Pi\left(\Sigma\Pi\right)^{[1]}$ circuits, the known proof techniques for lower bounds for $\Sigma\Pi\left(\Sigma\Pi\right)^{[1]}$ circuits (which were proved using *evaluation dimension* techniques of [24, 27]) seem to completely break down in this setting. In fact, Forbes [7] was able to confirm this, showing that there is a polynomial which is a sum of product of bivariates which has exponentially large evaluation dimension under all possible partitions of variables. Thus, studying this model seems like an interesting next step towards understanding non-homogeneous small depth algebraic computation. As far as we are aware there are also (not surprisingly) no nontrivial PIT results for the model. We are now ready to state our results.

1.1 Our results

Lower bounds

We show an exponential lower bound for the model of $\Sigma\Pi\left(\Sigma\Pi\right)^{[s]}$, when s is at most N^{μ} for any constant $0 \leq \mu < 1$ (N is the number of variables). More precisely, we show the following.

▶ **Theorem 1.2.** *Let $\mathbb{F}$ be a field of characteristic zero and μ be any constant such that $0 \leq \mu < 1$. There exists a family $\{P_N\}$ of polynomials over $\mathbb{F}$ in* VNP*, where P_N is of degree n in $N = n^{O_\mu(1)}$ variables, such that for any representation of P_N of the form*

$$P_N = \sum_{i=1}^{T} \prod_{j=1}^{d} Q_{ij}$$

where each Q_{ij} is polynomial in at most $s = N^{\mu}$ variables, it must be true that

$$T \cdot d \geq n^{\Omega_\mu(\sqrt{n})}.$$

Given the depth reduction results of [13] and the observation mentioned earlier from [17], it is known that any asymptotic improvement in the exponent of the lower bound (even for $s = O(\sqrt{n})$) would imply VNP is different from VP.

As discussed in the introduction, even though this model seems a natural generalization of the model of sums of products of univariate polynomials, our lower bound technique is very different from those used in proving lower bounds for sums of products of univariates. Our lower bound proof is based on ideas developed in the course of investigating homogeneous depth four arithmetic circuits [15, 22].

Blackbox PIT

We also consider the problem of PIT for the model of $\Sigma\Pi\left(\Sigma\Pi\right)^{[s]}$ circuits. For general sums of products of even bivariate polynomials, this question seems quite difficult, and as of now we are not even able to obtain subexponential time PIT. However, as a consequence of our lower bounds and by suitably adapting hardness randomness tradeoffs for arithmetic circuits developed in [14] and [6], we are able to obtain PIT results in the setting where the top fan-in of the circuit is bounded, and when we have the promise that the circuit computes a polynomial of low individual degree.

Our understanding of blackbox PIT for depth four circuits is very limited, and the results known are in very restricted settings. Saraf and Volkovich [28] gave blackbox PIT algorithms for multilinear depth 4 circuits with bounded top fan-in. To the best of our knowledge, the idea in [28] does not extend to the case of non-multilinear depth 4 circuits, even when the individual degree of each of the variables is at most 2. Recently, Oliveira et al [5] gave a subexponential time blackbox PIT for all depth four multilinear circuits[4]. In the non-multilinear setting, Agrawal et al. [2] gave PIT algorithms for constant depth formulas in which the number of *occurences* of each variable is bounded. Without going into the technical details, we remark that the notion of *bounded occur* is a generalization of the well studied notion of bounded reads. The most closely related results to those in this paper that we are aware of are the recent papers of Gupta [12] and Mukhopadhyay [23], which give

[4] The running time increases with the size of the circuit, and in particular, it is subexponential time for polynomial sized depth four multilinear circuits.

blackbox PIT results for sums of products of low degree polynomials, where the top sum fan-in is bounded and the circuits satisfy certain algebraic geometric restrictions.

So, the question of getting PIT results for general depth four circuits (even with bounded top and bottom fan-in) remains wide open. For instance we still do not know any nontrivial PIT results for a sum of constant many products of degree 2 polynomials. Though we still don't know how to deal with this question, when we replace the polynomials of low degree with polynomials of few variables (but of arbitrarily large degree), then we are able to obtain quasipolynomial PIT results. There is one added caveat however, that the final polynomial computed needs to be of low individual degree (as seems necessary for PIT results obtained from the known hardness-randomness tradeoffs for bounded depth circuits [6]). We now formally state the theorem.

▶ **Theorem 1.3.** *Let c and μ be arbitrary constants such that $c > 0$ and $0 \leq \mu < 1/2$, and let $\mathbb{F}$ be a field of characteristic zero. Let $\mathcal{C}$ be the set of polynomials P in N variables and individual degree at most k over $\mathbb{F}$, with the property that P can be expressed as*

$$P = \sum_{i=1}^{T} \prod_{j=1}^{d} Q_{ij}$$

such that

1. $T < \log^c N$
2. $k < \log^c N$
3. $d < N^c$
4. *each Q_{ij} depends on at most N^μ variables*

Then, there exists a constant $\epsilon < 1$ dependent only on c and μ, such that there is a hitting set of size $\exp(N^\epsilon)$ for $\mathcal{C}$ which can be constructed in time $\exp(N^\epsilon)$.

Moreover, from our proof, it also follows that if each of polynomial Q_{ij} depends only on $\log^{O(1)} N$ variables, then both the size of the hitting set and the time to construct it, are upper bounded by a quasipolynomial function in N.

Independent work

In a simultaneous independent work, Kayal and Saha [18] employ very similar techniques and ideas to show an analog of Theorem 1.2 for the iterated matrix multiplication polynomial (an entry in the product of n generic matrices of dimension $\text{poly}(n) \times \text{poly}(n)$) when each of the polynomials Q_{ij} depends on at most $\sqrt{n}$ variables.

Organisation of the paper

We provide an overview of the proofs in Section 2. We describe some definitions and preliminaries in Section 3. We present the proof of the lower bound in Section 4. We describe the application to blackbox PIT in Section 5 and conclude with some open problems in Section 6.

2 Proof overview

In this section, we provide an overview of the main ideas in proofs of Theorem 1.2 and Theorem 1.3.

2.1 Overview of proof of Theorem 1.2

We restate Theorem 1.2 for the sake of clarity.

▶ **Theorem 1.2** (restated). *Let $\mathbb{F}$ be a field of characteristic zero and μ be any constant such that $0 \leq \mu < 1$. There exists a family $\{P_N\}$ of polynomials over $\mathbb{F}$ in* VNP*, where P_N is of degree n in $N = n^{O_\mu(1)}$ variables, such that for any representation of P_N of the form $P_N = \sum_{i=1}^{T} \prod_{j=1}^{d} Q_{ij}$ where each Q_{ij} is polynomial in at most $s = N^\mu$ variables, it must be true that*

$$T \cdot d \geq n^{\Omega_\mu(\sqrt{n})}.$$

The key difference between proving the above lower bound and the lower bounds for homogeneous depth four circuits is that the formal degree of the circuit in the above case could be much larger than the degree of the polynomial, which is n. In fact, even the fan-in of the product gates at level 2, that is d could be much larger than n. Therefore, a straightforward application of homogeneous depth four circuit lower bounds does not seem to work. Our proof is in two steps and at a high level follows the strategy of the lower bound for non-homogeneous depth three circuits with bounded bottom fan-in by Kayal and Saha [17] with some key differences.

- In the first step, we obtain another representation of P_N, as

$$P_N = \sum_{i=1}^{Td2^{O(\sqrt{n})}} \prod_{j=1}^{n} Q'_{ij}$$

  where every monomial in each of the Q'_{ij} has *support*[5] at most s, although each Q'_{ij} could now depend on all the variables. The key property that we have gained from this transformation is that the fan-in of the product gates at level two is bounded by n now, which is the degree of P_N. However, we have no bound on the degree of the Q'_{ij}. Moreover, we have blown up the top fan-in a bit, but we will be able to tolerate this loss if s is small.
- In the second step, the strategy can be seen in two stages. If μ was very small, say 0.001, then we could have taken advantage of the fact that in the representation obtained in the first step above, the product fan-in is at most n and the support of every monomial in each of the Q'_{ij} is small, to prove an upper bound on the dimension of the space of projected shifted partial derivatives of the above representation. Comparing this dimension with that of our hard polynomial gives us our lower bound. For larger values of μ, we use random restrictions to ensure that all the monomials of *large support* in Q'_{ij} are set to zero. At the end of such a procedure, we are back to the low support case. This step of the proof is closely along the lines of the proof of homogeneous depth four arithmetic circuit lower bounds in [15, 22] although in the present case, formal degree of the circuit could be as large as n^2, which is much larger than the degree of the polynomial P_N. For such large formal degrees, in general we do not even know lower bounds for non-homogeneous depth three circuits.

We would like to point out that the first step of the proof above is similar to the homogenization step in the proof of lower bounds for general depth three circuits with bounded bottom fan-in by Kayal and Saha [17]. The key difference is that while the circuit they obtain at the end of

[5] A monomial is said to have support support s if it depends on at most s distinct variables.

this step is a strictly homogeneous circuit of formal degree n, we are unable to get a similar structure. The complication stems from the fact that when Q_{ij} are not affine forms, they could contain monomials of varying degrees. In this case, it seems difficult to obtain a strict homogenization with a small blow up in size. We get around this deficiency by a more subtle analysis in the second step, where we show a lower bound for a circuit which has a formal degree much larger than the degree of the polynomial being computed, but has some added structure. This step critically uses that the fact that the product fan-in at level two of these circuits is at most n, and the support of every monomial in each of the Q'_{ij} is small.

2.2 Overview of proof of Theorem 1.3

We first restate Theorem 1.3.

▶ **Theorem 1.3** (restated). *Let c and μ be arbitrary constants such that $c > 0$ and $0 \leq \mu < 1/2$, and let $\mathbb{F}$ be a field of characteristic zero. Let $\mathcal{C}$ be the set of polynomials P in N variables and individual degree at most k over $\mathbb{F}$, with the property that P can be expressed as $P = \sum_{i=1}^{T} \prod_{j=1}^{d} Q_{ij}$ such that*

1. $T < \log^c N$
2. $k < \log^c N$
3. $d < N^c$
4. *each Q_{ij} depends on at most N^μ variables*

Then, there exists a constant $\epsilon < 1$ dependent only on c and μ, such that there is a hitting set of size $\exp(N^\epsilon)$ for $\mathcal{C}$ which can be constructed in time $\exp(N^\epsilon)$.

The construction of the hitting set is based on the well known idea of using hard functions for derandomization. Our goal is to reduce the number of variables from N to at most N^δ for some constant $\delta < 1$, while maintaining the zeroness/nonzeroness of the polynomial being tested [14, 6]. Once we have done this, we take a brute force hitting set of size $(\text{Degree} + 1)^{\text{Number of variables}}$ as given by Lemma 5.5. To reduce the number of variables, we use the framework introduced by Kabanets and Impagliazzo [14].

The key technical step of the proof is to show that for a non-zero polynomial P as defined above, if there exists a polynomial $f \in \mathbb{F}[X_1, X_2, \ldots, X_{i-1}, X_{i+1}, X_{i+2}, \ldots, X_N]$ such that $X_i - f$ divides P, then f can also be expressed as a sum of products of polynomials in few variables of reasonably small size. This step crucially uses a statement about complexity of roots of polynomials computed by low depth circuits from [6]. Therefore, if f is a polynomial which does not have a small representation as a sum of products of polynomials in *few* variables, then $X_i - f$ does not divide P. This observation guarantees that the construction of hitting sets from hard polynomials given by [14] works for this class of circuits.

3 Notation and Preliminaries

We now introduce some notation and preliminary notions that we use in the rest of the paper.

Computational model

In this work, we consider the model of sums of products of polynomials in few variables. More formally, we consider representations of polynomials P (degree n in $N = n^{O(1)}$ variables) in

the form

$$P = \sum_{i=1}^{T} \alpha_i \cdot \prod_{j=1}^{d} Q_{ij} \tag{2}$$

where each Q_{ij} is an arbitrary polynomial (of arbitrarily high degree) in at most s variables and each α_i is a field constant. We call this the model of $\Sigma\Pi\left(\Sigma\Pi\right)^{[s]}$ circuits. We use the quantity Td as a measure of the size of a $\Sigma\Pi\left(\Sigma\Pi\right)^{[s]}$ circuit. Without loss of generality, we can assume that the degree zero term in each of the Q_{ij} is either zero or one. If it is a non-zero constant other than 1, we can extract it out and absorb it in α_i. For each of the product gates, the fan-in could be different, but we can assume without loss of generality that all the product fan-ins are equal to d. Observe that the d could be much larger than the degree of the polynomial P. Throughout this paper, we will be working over a field of characteristic zero.

Some basic notations

1. For an integer i, we denote the set $\{1, 2, \ldots, i\}$ by $[i]$.
2. By $\overline{X}$, we mean the set of variables $\{X_1, X_2, \ldots, X_N\}$.
3. For a polynomial P and a positive integer i, we represent by $\mathsf{Hom}^i[P]$, the homogeneous component of P of degree equal to i. By $\mathsf{Hom}^{\leq i}[P]$ and $\mathsf{Hom}^{\geq i}[P]$, we represent the component of P of degree at most i and at least i respectively.
4. The support of a monomial α is the set of variables which appear with a non-zero exponent in α. We denote the size of the support of α by $\mathrm{Supp}(\alpha)$.
5. Throughout the paper, we say that a function $f(N)$ is subexponential in N if there exists a positive real number ϵ, such that $\epsilon < 1$ and for all N sufficiently large, $f(N) < \exp(N^{\epsilon})$.
6. We say that a function $f(N)$ is quasipolynomial in N if there exists a positive absolute constant c, such that for all N sufficiently large, $f(N) < \exp(\log^c N)$.
7. In this paper, we only consider layered arithmetic circuits and we will be counting levels from top to bottom, starting with the output gates being at level one.
8. By a $\Sigma\Pi\Sigma\wedge$ circuit, we refer to a depth four circuit with all the product gates at the lowest level being replaced by powering ($\wedge$) gates. Similarly, by a $\Sigma\Pi\Sigma \wedge \Sigma\Pi$ circuit, we mean a depth six circuit all of whose product gates at level four from the top are powering gates.

Hitting set

Let $\mathcal{C}$ be a set of polynomials in N variables over a field $\mathbb{F}$. Then, a set $\mathcal{H} \subseteq \mathbb{F}^N$ is said to be a *hitting set* for the class $\mathcal{C}$, if for every polynomial $P \in \mathcal{C}$ such that P is not the identically zero polynomial, there exists a $p \in \mathcal{H}$ such that $P(p) \neq 0$.

Elementary symmetric polynomials

For variables $\overline{X} = \{X_1, X_2, \ldots, X_N\}$ and any integer $0 \leq l \leq N$, the elementary symmetric polynomial of degree l on variables $\overline{X}$ is defined as

$$\mathsf{ESYM}_l(\overline{X}) = \sum_{S \subseteq [N], |S| = l} \prod_{j \in S} X_j \,.$$

Projected shifted partial derivatives

A key idea behind the recent progress on lower bounds is the notion of *shifted partial derivatives* introduced in [16]. In this paper, we use a variant of the measure, called projected shifted partial derivatives introduced in [15] and subsequently used in [22]. Although we never explicitly do any calculations with the measure in this paper, we provide a brief introduction to it below since the bounds are based on it.

For a polynomial P and a monomial γ, $\partial_\gamma(P)$ is the partial derivative of P with respect to γ. For every polynomial P and a set of monomials $\mathcal{M}$, $\partial_{\mathcal{M}}(P)$ is the set of partial derivatives of P with respect to monomials in $\mathcal{M}$. The space of $(\mathcal{M}, m)$-projected shifted partial derivatives of a polynomial P is defined below.

▶ **Definition 3.1** ($(\mathcal{M}, m)$-projected shifted partial derivatives)**.** For an N variate polynomial $P \in \mathbb{F}[X_1, X_2, \ldots, X_N]$, set of monomials $\mathcal{M}$ and a positive integer $m \geq 0$, the space of $(\mathcal{M}, m)$-projected shifted partial derivatives of P is defined as

$$\langle \partial_{\mathcal{M}}(P) \rangle_m \stackrel{def}{=} \mathbb{F}\text{-}span\{\sigma(\prod_{i \in S} X_i \cdot g) : g \in \partial_{\mathcal{M}}(P), S \subseteq [N], |S| = m\} \tag{3}$$

Here, $\sigma(P)$ of a polynomial P is the projection of P on the multilinear monomials in its support. The measure of complexity of a polynomial that we use in this paper, is the dimension of projected shifted partial derivative space of P with respect to some set of monomials $\mathcal{M}$ and a parameter m. Formally,

$$\Phi_{\mathcal{M},m}(P) = \dim(\langle \partial_{\mathcal{M}}(P) \rangle_m) .$$

From the definitions, it is straight forward to see that the measure is subadditive.

▶ **Lemma 3.2** (Sub-additivity)**.** *Let P and Q be any two multivariate polynomials in $\mathbb{F}[X_1, X_2, \ldots, X_N]$. Let $\mathcal{M}$ be any set of monomials and m be any positive integer. Then, for all scalars α and β*

$$\Phi_{\mathcal{M},m}(\alpha \cdot P + \beta \cdot Q) \leq \Phi_{\mathcal{M},m}(P) + \Phi_{\mathcal{M},m}(Q) .$$

Approximations

We will refer to the following lemma to approximate expressions during our calculations.

▶ **Lemma 3.3** ([11])**.** *Let $a(n), f(n), g(n) : \mathbb{Z}_{>0} \to \mathbb{Z}_{>0}$ be integer valued functions such that $(f + g) = o(a)$. Then,*

$$\log \frac{(a+f)!}{(a-g)!} = (f+g)\log a \pm O\left(\frac{(f+g)^2}{a}\right) .$$

In the proofs in this paper, we use Lemma 3.3 only in situations where $(f+g)^2$ will be $O(a)$. In this case, the error term will be bounded by an absolute constant. So, up to constant factors, $\frac{(a+f)!}{(a-g)!} = a^{(f+g)}$. We use the symbol $\approx$ to indicate equality up to constant factors.

Complexity of coefficients and homogeneous components

We now summarise two simple lemmas which are useful for our proof. The first lemma summarises that given a circuit C for a polynomial $P \in \mathbb{F}[X1, X2, \ldots, X_N, Y]$ of degree at most d, for every $0 \leq i \leq d$, the coefficient of Y^i in P (when viewing P as a polynomial in $\mathbb{F}[X_1, X_2, \ldots, X_N][Y]$) can also be computed by a circuit of size not much larger than the size of C.

▶ **Lemma 3.4.** *Let $P \in \mathbb{F}[X_1, X_2, \ldots, X_N, Y]$ be a polynomial of degree at most d in Y over a field $\mathbb{F}$ of characteristic zero, such that P is computable by an arithmetic circuit C of size $|C|$. Let*

$$P = \sum_{i=0}^{d} Q_i(X_1, X_2, \ldots, X_N) \cdot Y^i$$

for polynomials $Q_i(X_1, X_2, \ldots, X_N) \in \mathbb{F}[X_1, X_2, \ldots, X_N]$. Then, for every i such that $0 \leq i \leq d$, the polynomial Q_i can be computed by an arithmetic circuit C' of size at most $|C| \cdot (d+1)$. Moreover, if the output gate of C is a $+$ gate, then the depth of C' is equal to the depth of C. Else, the depth of C' is at most 1 more than the depth of C.

Proof. We can view P as a univariate polynomial of degree at most d in Y with the coefficients coming from $\mathbb{F}(\overline{X})$. From the classical Lagrange interpolation, we know that the coefficient of Y^i in P can be written as an $\mathbb{F}(\overline{X})$ linear combination of the evaluations of P at $d+1$ distinct values of Y taken from $\mathbb{F}(\overline{X})$. In fact, more strongly, we can evaluate P at $d+1$ values of Y all chosen from $\mathbb{F}$ itself, in which case the constants in the linear combination are also from $\mathbb{F}$. So, Q_i can be computed by a circuit obtained from taking $d+1$ circuits each obtained from P by substituting Y by a scalar in $\mathbb{F}$, and taking their linear combination. Let this circuit be C'. Clearly the size of C' is at most $(d+1)$ times the size of C. If the output gate of C was an addition gate, then the outer addition for the linear combination can be absorbed into it, and the depth remains the same. Else, the depth increases by one. ◀

The second lemma stated below essentially says that the circuit complexity of homogeneous components of a polynomial is not much larger than the circuit complexity of the polynomial itself.

▶ **Lemma 3.5.** *Let P be a polynomial of degree at most d in N variables over a field $\mathbb{F}$ of characteristic zero, such that P is computable by an arithmetic circuit C of size $|C|$. Then, for every i such that $0 \leq i \leq d$, the homogeneous component of degree i of P can be computed by an arithmetic circuit C' of size at most $|C| \cdot (d+1)$. Moreover, if the output gate of C is a $+$ gate, then the depth of C' is equal to the depth of C. Else, the depth of C' is at most 1 more than the depth of C.*

Proof. Let $P'(t)$ be the polynomial obtained from P by replacing every variable X in P by $X \cdot t$ for a new variable t. We can view P' to be a univariate polynomial of degree at most d in t with the coefficients coming from $\mathbb{F}(\overline{X})$. Observe that for every i such that $0 \leq i \leq d$, the homogeneous component of P of degree equal to i is equal to the coefficient of t^i in P'. The proof now follows from Lemma 3.4. ◀

4 Proof of the lower bound

In this section, we give the proof of Theorem 1.2. We prove the lower bound for a variant of the well known family of Nisan-Wigderson polynomials defined by Kayal and Saha [17].

4.1 Target polynomials for the lower bound

We now define the family of polynomials of degree n in N variables for which we prove the lower bounds. The family is a variant of the Nisan-Wigderson polynomials which were introduced by Kayal et al in [19] in the context of lower bounds for homogeneous depth four circuits. The particular variant we use in the paper is due to Kayal and Saha [17].

The tradeoff between the number of variables N and the degree n will be parameterized by the parameter μ where $0 \leq \mu < 1$. First we need some parameters, which we define below.

1. $\delta = (1-\mu)/2$ is a positive real number such that $\mu + \delta < 1$.
2. $\gamma = \frac{2(\mu+\delta)+1}{1-\mu-\delta}$.
3. N is chosen such that N/n is a prime number between $n^{1+\gamma}$ and $2n^{1+\gamma}$. Such a prime number always exists from the Bertrand-Chebychev theorem. Without loss of generality, we pick the smallest one.
4. $\rho = (\mu+\delta)\frac{\log N}{\log n}$
5. $D = \frac{\gamma+\rho}{2(1+\gamma)} \cdot n$, where $D-1$ is the degree of the underlying univariate polynomials in the definition of $NW_{n,\mu}$.

Let ψ be the prime number equalling N/n. We are now ready to restate the definition of $NW_{n,\mu}$ from [17].

▶ **Definition 4.1** (Nisan-Wigderson Polynomials [17]). Let μ be a real number such that $0 \leq \mu < 1$. For a given μ and n, let N, D, ψ be as defined above. For the set of N variables $\{X_{ij} : i \in [n], j \in [\psi]\}$, we define the degree n homogeneous polynomial $NW_{n,\mu}$ as

$$NW_{n,\mu} = \sum_{\substack{f(z)\in\mathbb{F}_\psi[z] \\ deg(f)\leq D-1}} \prod_{i\in[n]} X_{if(i)} \,.$$

From the definition, we can observe the following properties of $NW_{n,\mu}$.

1. The number of monomials in $NW_{n,\mu}$ is exactly $\psi^D = n^{O(D)}$.
2. Each of the monomials in $NW_{n,\mu}$ is multilinear.
3. Each monomial corresponds to evaluations of a univariate polynomial of degree at most $D-1$ at all points of $\mathbb{F}_\psi$. Thus, any two distinct monomials agree in at most $D-1$ variables in their support.

We will also need the following lemma in our proof.

▶ **Lemma 4.2.** *Let μ be a non-negative real number less than 1. Given $q \in \mathbb{F}^N$, μ, n, we can evaluate the polynomial $NW_{n,\mu}$ at q in time $N^{O(n)}$.*

Proof. Given n and μ, we first find D, ψ as given by the choice of parameters. Once we have D, we iterate through every monomial α of degree n in the $\overline{X}$ variables which is supported on all the rows of the variable matrix and check if it is in the polynomial $NW_{n,\mu}$ by trying to find a univariate polynomial $f(z) \in \mathbb{F}_\psi[z]$ such that degree of f is at most $D-1$ and $\prod_{i\in[n]} X_{if(i)} = \alpha$. The interpolation takes only $\mathrm{Poly}(n)$ time, and the total number of monomials to try is at most N^n. So, we get the lemma. ◀

We now proceed with the proof as outlined in Section 2.1.

4.2 Reducing the product fan-in at level two

Let P be a homogeneous polynomial in N variables of degree n which has a $\Sigma\Pi\,(\Sigma\Pi)^{[s]}$ circuit of top fan-in T and product fan-in d at the second level. In other words, there exist polynomials $\{Q_{ij} : i \in [T], j \in [d]\}$ in at most s variables each, such that

$$P = \sum_{i=1}^{T} \alpha_i \cdot \prod_{j=1}^{d} Q_{ij} \,. \tag{4}$$

Recall that without loss of generality, we can assume that the constant term in each of the Q_{ij} is either 0 or 1. We have the following lemma.

▶ **Lemma 4.3.** *Let $\mathbb{F}$ be a field of characteristic zero. Let P be a homogeneous polynomial of degree n in N variables over $\mathbb{F}$ as defined above. For each i, $1 \leq i \leq T$ define the set*

$$S_i = \{j : 1 \leq j \leq d \text{ and } \mathsf{Hom}^0[Q_{ij}] = 1\}.$$

Then,

$$P = \sum_{i=1}^{T} \alpha_i \cdot \mathsf{Hom}^n \left[\prod_{j \notin S_i} Q_{ij} \times \sum_{l=0}^{n} \mathsf{ESYM}_l(\{\mathsf{Hom}^{\geq 1}[Q_{ij}] : j \in S_i\}) \right]. \tag{5}$$

Proof. To prove the lemma, we will try to extract out the homogeneous part of degree n of each product gate $\prod_{j=1}^{d} Q_{ij}$. Together with the fact that the polynomial P is homogeneous of degree n, we get the lemma. Every Q_{ij} with a non-zero constant term can be written as $\mathsf{Hom}^{\geq 1}[Q_{ij}] + 1$, since the constant term in each Q_{ij} is either 0 or 1. Now,

$$\prod_{j=1}^{d} Q_{ij} = \prod_{j \notin S_i} Q_{ij} \times \prod_{j \in S_i} (\mathsf{Hom}^{\geq 1}[Q_{ij}] + 1). \tag{6}$$

Decomposing the product $\prod_{j \in S_i} (\mathsf{Hom}^{\geq 1}[Q_{ij}] + 1)$ further, we have

$$\prod_{j \in S_i} (\mathsf{Hom}^{\geq 1}[Q_{ij}] + 1] = \sum_{l=0}^{|S_i|} \sum_{U \subseteq S_i : |U| = l} \prod_{j \in U} \mathsf{Hom}^{\geq 1}[Q_{ij}]. \tag{7}$$

Now, observe that the degree of every monomial in $\prod_{j \in U} \mathsf{Hom}^{\geq 1}[Q_{ij}]$ is at least as large as the size of U. So, for every subset U of size larger than n, $\prod_{j \in U} \mathsf{Hom}^{\geq 1}[Q_{ij}]$ is a polynomial of degree strictly larger than n. Also, for any fixed l, the expression $\sum_{U \subseteq S_i : |U| = l} \prod_{j \in U} \mathsf{Hom}^{\geq 1}[Q_{ij}]$ is precisely the elementary symmetric polynomial of degree l in the set of variables $\{\mathsf{Hom}^{\geq 1}[Q_{ij}] : j \in S_i\}$. Therefore,

$$\mathsf{Hom}^{\leq n} \left[\prod_{j \in S_i} (\mathsf{Hom}^{\geq 1}[Q_{ij}] + 1) \right] = \mathsf{Hom}^{\leq n} \left[\sum_{l=0}^{n} \mathsf{ESYM}_l(\{\mathsf{Hom}^{\geq 1}[Q_{ij}] : j \in S_i\}) \right]. \tag{8}$$

Therefore,

$$\mathsf{Hom}^n \left[\prod_{j=1}^{d} Q_{ij} \right] = \mathsf{Hom}^n \left[\prod_{j \notin S_i} Q_{ij} \times \sum_{l=0}^{n} \mathsf{ESYM}_l(\{\mathsf{Hom}^{\geq 1}[Q_{ij}] : j \in S_i\}) \right]. \tag{9}$$

Summing up for all i, we get the lemma. ◀

The lemma above has in some sense helped us locate the monomials of degree n in the circuit, which otherwise has a much higher formal degree. We now combine the above lemma with the well known fact that elementary symmetric polynomial of degree l in k variables can be computed by homogeneous $\Sigma\Pi\Sigma\wedge$ circuits of size at most $k2^{O(\sqrt{l})}$ to obtain a $\Sigma\Pi\Sigma \wedge \Sigma\Pi$ circut C' such that the fan-in of the product gates at level two is at most n. We use the following theorem (Theorem 5.2) by Shpilka and Wigderson [31].

▶ **Theorem 4.4** (Shpilka-Wigderson [31])**.** *For every set of variables $\{Y_1, Y_2, \ldots, Y_m\}$ and a positive integer l, $\mathsf{ESYM}_l(\{Y_1, Y_2, \ldots, Y_m\})$ can be computed by a homogeneous $\Sigma\Pi\Sigma\wedge$ circuit of size $m2^{O(\sqrt{l})}$.*

We now prove the following lemma.

▶ **Lemma 4.5.** *Let $\mathbb{F}$ be a field of characteristic zero. Let P be a polynomial of degree n in N variables over $\mathbb{F}$ which is computable by an $\Sigma\Pi(\Sigma\Pi)^{[s]}$ circuit C of top fan-in T and the degree of product gates at level two being d. So, P can be represented as*

$$P = \sum_{i=1}^{T} \alpha_i \cdot \prod_{j=1}^{d} Q_{ij} .$$

Then, P can be represented as the homogeneous component of degree n of a polynomial computed by a $\Sigma\Pi\Sigma \wedge \Sigma\Pi$ circuit C'' with the following properties :

1. *The inputs to the $\wedge$ gates are the polynomials $\{\mathsf{Hom}^{\geq 1}[Q_{ij}] : 1 \leq i \leq T, 1 \leq j \leq d\}$*
2. *The fan-in of the $\times$ gates at the second level from the top is at most n*
3. *The top fan-in of C'' is at most $Tdn2^{O(\sqrt{n})}$.*

Proof. From Lemma 4.3, we know that for the set S_i defined as

$$S_i = \{j : 1 \leq j \leq d \text{ and } \mathsf{Hom}^0[Q_{ij}] = 1\}$$

the polynomial P can be written as

$$P = \sum_{i=1}^{T} \alpha_i \cdot \mathsf{Hom}^n \left[\prod_{j \notin S_i} Q_{ij} \times \sum_{l=0}^{n} \mathsf{ESYM}_l(\{\mathsf{Hom}^{\geq 1}[Q_{ij}] : j \in S_i\}) \right]$$

which is the same as

$$P = \mathsf{Hom}^n \left[\sum_{i=1}^{T} \alpha_i \cdot \prod_{j \notin S_i} Q_{ij} \times \sum_{l=0}^{n} \mathsf{ESYM}_l(\{\mathsf{Hom}^{\geq 1}[Q_{ij}] : j \in S_i\}) \right] .$$

Observe that the polynomial $\prod_{j \notin S_i} Q_{ij}$ has degree at least $d - |S_i|$. We remark that if $d - |S_i|$ is larger than n, then such product gates do not contribute anything to the degree n component of the polynomial and hence can be discarded without loss of generality; hence we assume $n - (d - |S_i|) > 0$. So, we could confine the inner sum from $l = 0$ to $l = n - (d - |S_i|)$, and still preserve the degree n part of the polynomial, which is what we are interested in. From Theorem 4.4, we know that for every $0 \leq l \leq n$, we can compute the polynomial $\mathsf{ESYM}_l(\{\mathsf{Hom}^{\geq 1}[Q_{ij}] : j \in S_i\})$ by a $\Sigma\Pi\Sigma\wedge$ circuit of top fan-in at most $d \times 2^{O(\sqrt{l})}$ which takes as input the polynomials $\{\mathsf{Hom}^{\geq 1}(Q_{ij}) : 1 \leq j \leq d\}$. From the homogeneity of the circuits given by Theorem 4.4, it follows that the product gates at level two of these circuits have fan-in at most the degree of polynomial they compute, which is at most $n - (d - |S_i|)$. So, it follows that the polynomial

$$\tilde{P} = \left(\sum_{i=1}^{T} \alpha_i \cdot \prod_{j \notin S_i} Q_{ij} \times \sum_{l=0}^{n-(d-|S_i|)} \mathsf{ESYM}_l(\{\mathsf{Hom}^{\geq 1}[Q_{ij}] : j \in S_i\}) \right)$$

can be computed by a $\Sigma\Pi\Sigma \wedge \Sigma\Pi$ circuit, with top fan-in at most $Tdn \cdot 2^{O(\sqrt{n})}$, which satisfies the conditions in the lemma. ◀

Finally, given the circuit C'' constructed above, we can construct a circuit which computes the polynomial P as given by Lemma 3.5. For this, observe that the monomials of degree strictly larger than n in any of the Q_{ij} do not contribute to degree n part of $\tilde{P}$. So, we can

drop them, while still preserving the degree n part of $\tilde{P}$. Therefore, the degree of $\tilde{P}$ can be upper bounded by n^2d. We can recover the degree n part of $\tilde{P}$ by interpolation which blows up the top fan-in by a factor of at most n^2d.

In this process, the fan-in of the product gates at level two remains unchanged. Strictly speaking, inputs to the powering gate $\wedge$ at level four may no longer be the polynomials $\mathsf{Hom}^{\geq 1}[Q_{ij}]$, since in the process of interpolation, we replaced every variable X_i by $X_i.t$ in $\tilde{P}$ and looked at the resulting polynomial $\tilde{P}'$ as a univariate polynomial in t over the function field $\mathbb{F}(\overline{X})$. We then evaluated $\tilde{P}'$ at sufficiently many values of $t \in \mathbb{F}$ and then took their $\mathbb{F}$ linear combination. So, each of the polynomials $\mathsf{Hom}^{\geq 1}[Q_{ij}]$ gives rise to many other polynomials, one each for different values of t. We will call them the *siblings* of $\mathsf{Hom}^{\geq 1}[Q_{ij}]$. The key observation for our proof is that the set of variables in the siblings of $\mathsf{Hom}^{\geq 1}[Q_{ij}]$ is the same as the set of variables in $\mathsf{Hom}^{\geq 1}[Q_{ij}]$. From the lemma and the discussion above, we have the following corollary.

▶ **Corollary 4.6.** *Let $\mathbb{F}$ be a field of characteristic zero. Let P be a polynomial of degree n in N variables over $\mathbb{F}$ which is computable by an $\Sigma\Pi\left(\Sigma\Pi\right)^{[s]}$ circuit C of top fan-in T and the degree of product gates at level two being d. So, P can be represented as*

$$P = \sum_{i=1}^{T} \alpha_i \cdot \prod_{j=1}^{d} Q_{ij}\,.$$

Then, P can be computed by a $\Sigma\Pi\Sigma \wedge \Sigma\Pi$ circuit C'' with the following properties :

1. *The inputs to the $\wedge$ gates are the siblings of polynomials $\{\mathsf{Hom}^{\geq 1}[Q_{ij}] : 1 \leq i \leq T, 1 \leq j \leq d\}$*
2. *The fan-in of the $\times$ gates at the second level from the top is at most n*
3. *The top fan-in of C'' is at most $Td^2n^3 2^{O(\sqrt{n})}$.*

4.3 Random Restrictions

From the definition, it follows that the total number of variables in $NW_{n,\mu}$ is N. Let the set of all these variables be $\mathcal{V}$. We now define our random restriction procedure by defining a distribution $\mathcal{D}$ over subsets $V \subset \mathcal{V}$. The random restriction procedure will sample $V \leftarrow \mathcal{D}$ and then keep only those variables "alive" that come from V and set the rest to zero. We will denote the restriction of the polynomial obtained by such a restriction as $NW_{n,\mu}|_V$. Observe that a random restriction also results in a distribution over all circuits computing the polynomial $NW_{n,\mu}$. We denote by $C|_V$ the restriction of a circuit C obtained by setting every input gate in C which is labelled by a variable outside V to 0.

The distribution $\mathcal{D}_p$: Each variable in $\mathcal{V}$ is independently kept alive with a probability p. We will choose the value of p based on the parameter μ.

4.4 Analysing the circuit under random restrictions

Let C be a $\Sigma\Pi\left(\Sigma\Pi\right)^{[N^\mu]}$ circuit computing the polynomial $NW_{n,\mu}$. Let the top fan-in of C be T and the product fan-in at the second level be d. So, we have the following expression:

$$NW_{n,\mu} = \sum_{i=1}^{T} \alpha_i \cdot \prod_{j=1}^{d} Q_{ij}$$

where each Q_{ij} depends on at most N^μ variables.

Recall that from the choice of parameters $\delta = (1-\mu)/2$. Let s be a parameter, which we later set such that $s = \Theta(\sqrt{n})$. If $T \cdot d \geq N^{\frac{\delta}{4}s}$, then we already have the desired lower bound of $n^{\Omega(\sqrt{n})}$ on the size of C and we are done. Therefore, for the rest of this discussion, we will assume that $T \cdot d \leq N^{\frac{\delta}{4}s}$. We now apply the transformation to C given by Corollary 4.6 to obtain a $\Sigma\Pi\Sigma \wedge \Sigma\Pi$ circuit C'', which has the following properties:

1. The inputs to the $\wedge$ gates are the siblings of polynomials $\{\mathsf{Hom}^{\geq 1}[Q_{ij}] : 1 \leq i \leq T, 1 \leq j \leq d\}$
2. The fan-in of the $\times$ gates at the second level from the top is at most n
3. The top fan-in of C'' is at most $Td^2n^3 2^{O(\sqrt{n})}$.

We now analyse the effect of the random restrictions on the circuit C''. We will choose a parameter $p = N^{-\mu-\delta}$ and keep every variable alive with a probability p. The circuit C'' can be represented as

$$C'' = \sum_u \prod_v D_{uv} \,.$$

Here, each D_{uv} is a sum of powers of the siblings of $\mathsf{Hom}^{\geq 1}[Q_{ij}]$. Our goal is to argue that under random restrictions, all the monomials in each of the D_{uv} are of small support (support at most s).

For any polynomial P in N^μ variables and any integers t, t_0 such that $t_0 < t$, observe that P^t can be written as

$$P^t = P_0 + \sum_\alpha \alpha \cdot P_\alpha$$

where P_0 is the part of P consisting of monomials of support strictly less than t_0. The inner sum is over all multilinear monomials α of support equal to t_0. Such a decomposition may not be unique, but for this application, it would suffice to work with any one such decomposition. The number of such monomials α is at most $\binom{N^\mu}{t_0}$. The probability that one such monomial survives the random restriction procedure is equal to p^{t_0}. So, the expected number of such multilinear monomials α surviving the random restriction procedure is at most $\binom{N^\mu}{t_0} \cdot p^{t_0}$. The crucial observation is that if no such monomials survive, then only the monomials in P_0 survive, all of which have support at most $t_0 - 1$.

Now, observe that each of the D_{uv} are a sum of powers of the siblings of polynomials in the set $\{\mathsf{Hom}^{\geq 1}[Q_{ij}] : 1 \leq i \leq T, 1 \leq j \leq d\}$. Define $\mathcal{B}$ to be the set of all multilinear monomials of support equal to s, supported entirely on variables in any of the polynomials Q_{ij} for some $1 \leq i \leq T, 1 \leq j \leq d$. From the discussion in the paragraph above, the following observation follows.

▶ **Observation 4.7.** *Let the polynomials D_{uv}, Q_{ij} and the set $\mathcal{B}$ be as defined above. Then,*

- $|\mathcal{B}| \leq T \cdot d \cdot \binom{N^\mu}{s}$
- *If none of the monomials in $\mathcal{B}$ survive under some random restrictions, then each of the polynomials D'_{uv} obtained as a restriction of D_{uv} has all monomials of support at most s.*

Proof. The bound on the size trivially follows from the fact that each of the Q_{ij} depends on at most N^μ variables. For the second item, observe that each of the D_{uv} is a sum of powers of siblings of the $\mathsf{Hom}^{\geq 1}[Q_{ij}]$ and all the siblings are supported on the same set of variables. If all the monomials in the set $\mathcal{B}$ are set to zero, then the surviving monomials in any power of any of the siblings of $\mathsf{Hom}^{\geq 1}[Q_{ij}]$ has support at most s. ◀

We now estimate the probability that at least one of the monomials in the set $\mathcal{B}$ survives the random restriction procedure. We have the following lemma.

▶ **Lemma 4.8.** *Let δ be a positive real number such that $\delta = (1-\mu)/2$ and let $p = N^{-\mu-\delta}$. Then*

$$Pr_{V \leftarrow \mathcal{D}_p}\left[|\mathcal{B}|_V| \geq 1\right] \leq N^{-3/4\cdot\delta\cdot s}\,.$$

Proof. We know that

$$|\mathcal{B}| \leq T \cdot d \cdot \binom{N^{\mu}}{s}$$

and the probability that any fixed monomial in $\mathcal{B}$ survives the random restriction procedure is at most p^s. So

$$\mathbb{E}_{V \leftarrow \mathcal{D}_p}[|\mathcal{B}_V|] \leq T \cdot d \cdot \binom{N^{\mu}}{s} \cdot p^s\,.$$

Now, observing that the value of $T \cdot d$ is at most $N^{\frac{\delta}{4}s}$ and $p = N^{-\mu-\delta}$, the expected value is at most

$$N^{\frac{\delta}{4}s}\binom{N^{\mu}}{s} \cdot N^{-(\mu+\delta)s} \leq N^{-3/4\cdot\delta\cdot s}\,.$$

The lemma then follows by Markov's inequality. ◀

As a corollary of Lemma 4.8 and Observation 4.7, we get the following lemma.

▶ **Lemma 4.9.** *Let δ be a positive real number such that $\delta = (1-\mu)/2$ and let $p = N^{-\mu-\delta}$. Then with probability at least $1 - N^{-3/4\cdot\delta\cdot s}$ over random restrictions $V \leftarrow \mathcal{D}_p$, the polynomial computed by the circuit $C''|_V$ can be written as $\sum_{u=1}^{T'}\prod_{v=1}^{n} D'_{uv}$, where each of the monomials in each of the polynomials D'_{uv} has support at most s.*

4.5 Upper bound on the complexity of C

In order to upper bound the dimension of the projected shifted partial derivatives (under random restrictions) of the $\Sigma\Pi\,(\Sigma\Pi)^{[s]}$ circuit C, Corollary 4.6 implies that it suffices to upper bound the dimension of the space of projected shifted partial derivatives of the $\Sigma\Pi\Sigma \wedge \Sigma\Pi$ circuit C'' given by Corollary 4.6. In some sense, C'' is more structured than C and this lets us prove a better upper bound.

Recall that we are under the assumption that for the circuit C, the product of the top fan-in and the product fan-in at level two is at most $N^{\frac{\delta}{4}\cdot s}$, else we are already done. From Lemma 4.9, we know that with a high probability, under random restrictions, we are left with a circuit of the form $\sum_{u=1}^{T'}\prod_{v=1}^{n} D'_{uv}$ where each of the monomials in each of the polynomials D'_{uv} has support at most s. The upper bound on the complexity of the projected shifted partial derivatives of $\sum_{u=1}^{T'}\prod_{v=1}^{n} D'_{uv}$ then just follows from the upper bound for homogeneous depth four circuits of bounded bottom support proved in [15, 22]. We restate the bound from [22].

▶ **Lemma 4.10.** *Let C be a depth 4 circuit with the fan-in or product gates at level two bounded by n, the bottom support bounded by s and computing a polynomial in N variables. Let $\mathcal{M}$ be a set of monomials of degree equal to r and let m be a positive integer. Then,*

$$\Phi_{\mathcal{M},m}(C) \leq \textit{Top fan-in}(C)\binom{n+r}{r}\binom{N}{m+rs}$$

for any choice of m, r, s, N satisfying $m + rs \leq N/2$.

The upper bound for $\Sigma\Pi(\Sigma\Pi)^{[N^\mu]}$ circuits, follows easily form the above lemma after random restrictions, and we formalize this in the lemma below.

▶ **Lemma 4.11.** *Let μ be a positive real number such that $0 \leq \mu < 1$. Let $\delta = (1-\mu)/2$ and let $p = N^{-\mu-\delta}$ and let $\mathbb{F}$ be a field of characteristic zero. Let P be a polynomial of degree n in N variables over $\mathbb{F}$ which is computed by an $\Sigma\Pi(\Sigma\Pi)^{[N^\mu]}$ circuit C of top fan-in T and degree of product gates at level two at most d, i.e P can represented as*

$$P = \sum_{i=1}^{T} \alpha_i \cdot \prod_{j=1}^{d} Q_{ij}$$

where α_i are field constants. Let m and r be positive integers satisfying $m + rs \leq N/2$ and $\mathcal{M}$ be any subset of multilinear monomials of degree equal to r. If $Td \leq N^{\frac{s\cdot\delta}{4}}$, then with probability at least $1 - N^{-3/4\cdot\delta\cdot s}$ over random restrictions $V \leftarrow \mathcal{D}_p$,

$$\Phi_{\mathcal{M},m}(C|_V) \leq Td^2n^3 \cdot rs \cdot 2^{O(\sqrt{n})} \cdot \binom{N}{m+rs} \cdot \binom{n+r}{r}.$$

Proof. The lemma follows immediately from Corollary 4.6, Lemma 4.9 and Lemma 4.10. ◀

4.6 Nisan-Wigderson polynomial under random restrictions

To complete the proof of Theorem 1.2, we need a lower bound on the dimension of the space of projected shifted partial derivatives of the polynomial $NW_{n,\mu}$, under random restrictions. To this end, we will use the lower bound proved by Kayal and Saha [17]. We first enumerate our choice of parameters. Recall that $\delta = (1-\mu)/2$ is a positive real number.

1. $\gamma = \frac{2(\mu+\delta)+1}{1-\mu-\delta}$
2. N is such that N/n is set equal to the smallest prime number between $n^{1+\gamma}$ and $2n^{1+\gamma}$.
3. $\rho = (\mu+\delta)\frac{\log N}{\log n}$
4. $D = \frac{\gamma+\rho}{2(1+\gamma)} \cdot n$, where $D-1$ is the degree of the underlying univariate polynomials in the definition of $NW_{n,\mu}$.
5. r, s which are the order of derivative and the bound on bottom support of the circuit after random restrictions respectively, are chosen such that $r = \epsilon_1 \cdot \sqrt{n}, s = \epsilon_2 \cdot \sqrt{n}$. Here, ϵ_1 and ϵ_2 are small enough positive real numbers satisfying $\epsilon_1 \cdot \epsilon_2 = 0.001n$.
6. $m = \frac{N}{2}(1 - r\frac{\ln n}{n})$ is the degree of the shifts.
7. $p = N^{-(\mu+\delta)}$ is the probability with which each variable is independently kept alive.
8. $\mathcal{M}$ is the set of all multilinear monomials of degree r. We take partial derivatives with respect to monomials in this set.

We are now ready to state the lower bound on the dimension of projected shifted partial derivatives as in [17].

▶ **Lemma 4.12** (Kayal-Saha [17]). *Let $NW_{n,\mu}$ be Nisan-Wigderson polynomials as defined in Definition 4.1. Let $\mathbb{F}$ be any field of characteristic zero. Then, for the choice of parameters defined above*

$$\Phi_{\mathcal{M},m}(NW_{n,\mu}|_V) \geq \frac{1}{n^{O(1)}} min\left(\frac{p^r}{4^r} \cdot \binom{N}{r} \cdot \binom{N}{m}, \binom{N}{m+n-r}\right)$$

with probability at least $1 - \frac{1}{n^{\theta(1)}}$ over random restrictions $V \leftarrow \mathcal{D}_p$.

4.7 Wrapping up the proof of Theorem 1.2

From Lemma 4.12 and Lemma 4.9, we know that with a non-zero probability over the random restrictions V from the distribution $\mathcal{D}_p$, the following two conditions hold.

1. $\Phi_{\mathcal{M},m}(NW_{n,\mu}|_V) \geq \frac{1}{n^{O(1)}} \min\left(\frac{p^r}{4^r} \cdot \binom{N}{r} \cdot \binom{N}{m}, \binom{N}{m+n-r}\right)$.
2. $\Phi_{\mathcal{M},m}(C|_V) \leq Td^2n^3 \cdot rs \cdot 2^{O(\sqrt{n})} \cdot \binom{N}{m+rs} \cdot \binom{n+r}{r}$.

If C computed the polynomial $NW_{n,\mu}$, then

$$Td^2n^3 \cdot rs \geq \frac{\frac{1}{n^{O(1)}} \min\left(\frac{p^r}{4^r} \cdot \binom{N}{r} \cdot \binom{N}{m}, \binom{N}{m+n-r}\right)}{2^{O(\sqrt{n})} \cdot \binom{N}{m+rs} \cdot \binom{n+r}{r}} .$$

From the calculations in Appendix A, it follows that for our choice of parameters, the ratio is at least $\exp(\sqrt{n}\log n)$. So, we have the following theorem.

▶ **Theorem 4.13.** *Let μ be an absolute constant such that $0 \geq \mu < 1$ and $\mathbb{F}$ be a field of characteristic zero. For $1 \leq i \leq T$ and $1 \leq j \leq d$, if there exist polynomials Q_{ij}, each dependent on only $s = N^{\mu}$ variables, such that*

$$NW_{n,\mu} = \sum_{i=1}^{T} \prod_{j=1}^{d} Q_{ij} .$$

Then

$$T \cdot d \geq n^{\Omega_{\mu}(\sqrt{n})} .$$

As a remark, we mention here that the lower bound above also holds for any translation $NW_{n,\mu}(\overline{X} + \overline{a})$ of the polynomial $NW_{n,\mu}(\overline{X})$. This is because the highest degree term of $NW_{n,\mu}(\overline{X} + \overline{a})$ equals the polynomial $NW_{n,\mu}(\overline{X})$ and from Lemma 3.5, the homogeneous components of a polynomial computable by small sized $\Sigma\Pi\left(\Sigma\Pi\right)^{[s]}$ circuits also have small sized $\Sigma\Pi\left(\Sigma\Pi\right)^{[s]}$ circuits. We leave the details to the interested reader.

5 Application to polynomial identity testing

In this section, we prove Theorem 1.3. We are interested in identity testing for $\Sigma\Pi\left(\Sigma\Pi\right)^{[s]}$ circuits, i.e for polynomials in N variables $\{X_1, X_2, \ldots, X_N\}$ which can be expressed in the form

$$P = \sum_{i=1}^{T} \prod_{j=1}^{d} Q_{ij}$$

such that

1. The individual degree in P of every variable is at most k
2. Each Q_{ij} depends on at most s variables

For the case of this application, we will think of k, T being polynomial in $(\log N)$ and s being $N^{1/2-\epsilon}$ for a positive constant ϵ. Observe that the bound on individual degree lets us upper bound the total degree of the polynomials by Nk.

We describe the construction of the hitting set in Section 5.2 and prove its correctness in Section 5.3. We go over some preliminaries that we need in our proof in the next section.

5.1 Some preliminaries

In the following lemma, we prove some properties of the model of $\Sigma\Pi(\Sigma\Pi)^{[s]}$ circuits, which will be useful in the proof of the identity testing result.

▶ **Lemma 5.1.** *Let $\mathbb{F}$ be a field of characteristic zero. Let P be a non-zero polynomial in N variables and individual degree at most k over $\mathbb{F}$, which is computed by a $\Sigma\Pi(\Sigma\Pi)^{[s]}$ circuit C of top fan-in T and product fan-in d at level two, i.e P can be expressed as*

$$P = \sum_{i=1}^{T} \prod_{j=1}^{d} Q_{ij}$$

such that for each $i \in [T]$ and $j \in [d]$, Q_{ij} depends on at most s variables. Then, the following are true.

1. *For every variable y and integer $1 \leq j \leq k$, $\frac{\partial^j P}{\partial y^j}$ can be computed by a circuit of the form*

$$\frac{\partial^j P}{\partial y^j} = \sum_{i=1}^{T'} \prod_{j=1}^{d} Q'_{ij}$$

   where $T' \leq T \cdot (k+1)^2$ and each of the polynomials Q'_{ij} depends on at most s variables.
2. *For any $a \in \mathbb{F}^N$, $P(\overline{X} + \overline{a})$ can be computed by a circuit of the form*

$$P(\overline{X} + \overline{a}) = \sum_{i=1}^{T} \prod_{j=1}^{d} Q''_{ij}$$

   where each of the polynomials Q''_{ij} depends on at most s variables.

Proof. The proof of the second item is immediate from the definitions. The only thing that changes due to a translation is the number of monomials in the Q_{ij}. The number of variables that each Q_{ij} depends on remains unchanged, and so does the fan-in of the top sum gate and the product gates at level two.

We now prove the first item. Let the set of variables in P be $\overline{X} = \overline{X'} \cup \{y\}$ where X' is of size $N-1$. Since the individual degree of P is at most k, we can write $P = \sum_{i=0}^{k} C_i(\overline{X'}) \cdot y^i$. Here, $C_i(\overline{X'})$ are polynomials only in the X' variables and are the coefficient of y^i, when viewing P as an element of $\mathbb{F}[\overline{X'}][y]$. Now, for every $0 \leq i \leq k$, we can compute each of C_i by a $\Sigma\Pi(\Sigma\Pi)^{[s]}$ circuit with top fan-in at most $T \cdot (k+1)$ by interpolation as given by Lemma 3.4. All the partial derivatives of P with respect to y are linear combinations of the terms of the form $C_{j_1} \cdot y^{j_2}$. And so, the result follows. ◀

We will also need the following simple fact about polynomials.

▶ **Lemma 5.2.** *Let $\mathbb{F}$ be a field of characteristic zero. Let $R \in \mathbb{F}[y]$ be a non-zero polynomial of degree at most t over the field $\mathbb{F}$. Then, for every $a \in \mathbb{F}$ such that $R(a) = 0$, there exists a j such that $0 \leq j \leq t-1$ and $\frac{\partial^j R}{\partial y^j}(a) = 0$ and $\frac{\partial^{j+1} R}{\partial y^{j+1}}(a) \neq 0$.*

Proof. Let the degree of R in y be equal to t'. This means that the coefficient of highest degree term $y^{t'}$ in R is non-zero. Let us call the coefficient of $y^{t'}$ in $R(y)$ as $C_{t'}$. We know that $C_{t'}$ is nonzero. Consider $j = t' - 1$. The lemma immediately follows. ◀

We will crucially use the following result of Dvir, Shpilka, Yehudayoff [6] in the analysis of the hitting set constructed in this paper.

▶ **Lemma 5.3** (Dvir, Shpilka, Yehudayoff [6]). *For a field $\mathbb{F}$, let $P \in \mathbb{F}[X_1, X_2, \ldots, X_N, Y]$ be a non-zero polynomial of degree at most k in Y. Let $f \in \mathbb{F}[X_1, X_2, \ldots, X_N]$ be a polynomial such that $P(X_1, X_2, \ldots, X_N, f) = 0$ and $\frac{\partial P}{\partial Y}(0, 0, \ldots, 0, f(0, 0, \ldots, 0)) \neq 0$. Let*

$$P = \sum_{i=0}^{k} C_i(X_1, X_2, \ldots, X_N) \cdot y^i \,.$$

Then, for every $t \geq 0$, there exists a polynomial $R_t \in \mathbb{F}[Z_1, Z_2, \ldots, Z_{k+1}]$ of degree at most t such that

$$\mathsf{Hom}^{\leq t}[f(X_1, X_2, \ldots, X_N)] = \mathsf{Hom}^{\leq t}[R_t(C_0, C_1, \ldots, C_k)] \,.$$

A key technical idea in the proof will be the notion of Nisan-Wigderson designs introduced in [26]. We will use the following lemma.

▶ **Lemma 5.4** (Nisan-Wigderson [26]). *For every $a, b \in \mathbb{N}$, $b < 2^a$, there exists a family of sets $S_1, S_2, \ldots, S_b \subseteq \{1, 2, \ldots, l\}$ such that*
1. *$l \in O(a^2 / \log b)$*
2. *for all i, $|S_i| = a$*
3. *for all $i \neq j$, $|S_i \cap S_j| \leq \log b$*

Moreover, such a set family can be constructed in time polynomial in b and 2^l.

We will also use the following lemma of Alon [4] very crucially in our proof.

▶ **Lemma 5.5** (Combinatorial Nullstellensatz [4]). *Let P be a non-zero polynomial of individual degree at most d in N variables over a large enough field $\mathbb{F}$. Let S be an arbitrary subset of $\mathbb{F}$ of size $d + 1$. Then, there exists a point p in S^N such that $P(p) \neq 0$.*

5.2 Blackbox PIT for $\Sigma\Pi\,(\Sigma\Pi)^{[s]}$ circuits

In this section, we prove the following theorem.

▶ **Theorem 5.6.** *Let c and μ be arbitrary constants such that $c > 0$ and $0 \leq \mu < 1/2$, and let $\mathbb{F}$ be a field of characteristic zero. Let $\mathcal{C}$ be the set of polynomials P in N variables and individual degree at most k over $\mathbb{F}$, with the property that P can be expressed as*

$$P = \sum_{i=1}^{T} \prod_{j=1}^{d} Q_{ij}$$

such that
1. *$T < \log^c N$*
2. *$k < \log^c N$*
3. *$d < N^c$*
4. *each Q_{ij} depends on at most N^μ variables*

Then, there exists a constant $\epsilon < 1$ dependent only on c and μ, such that there is a hitting set of size $\exp(N^\epsilon)$ for $\mathcal{C}$ which can be constructed in time $\exp(N^\epsilon)$.

From our proof, it also follows that if each of polynomial Q_{ij} depends only on $\log^{O(1)} N$ variables, then both the size of the hitting set and the time to construct it, are upper bounded by a quasipolynomial function in N. In the rest of the section, we prove Theorem 5.6. We start by describing the construction of the hitting set $\mathcal{H}$.

5.2.1 Construction of hitting sets for $\Sigma\Pi(\Sigma\Pi)^{[N^\mu]}$ circuits for $0 \leq \mu < 1/2$

Given μ such that $0 \leq \mu < 1/2$, we pick the parameter μ' such that $0 < \mu' < 1$ and $\frac{2\mu}{\mu'}$ is a positive constant strictly smaller than 1. We construct a family of Nisan-Wigderson designs as described in Lemma 5.4 with the following parameters:

1. b, the number of sets is set equal to N.
2. a, the size of each of the sets S_i is set equal to $N^{\frac{\mu}{\mu'}} \log^{\frac{1}{\mu'}} N$.
3. l, the size of the universe is chosen large enough in order to satisfy the hypothesis of Lemma 5.4. From Lemma 5.4, it follows that we can pick l which is not too large ($l \in O(a^2/\log b)$). For the above chosen values of a, b, there is a choice of l such that l is at most $N^{\frac{2\mu}{\mu'}} \log^{\frac{2}{\mu'}-1} N$.

Recall that our goal is to construct a hitting set for $\Sigma\Pi(\Sigma\Pi)^{[N^\mu]}$ circuits. Observe that the choice of parameters l, a, b satisfy the hypothesis of Lemma 5.4. So, we get a collection of N subsets $S_1, S_2, \ldots, S_N$ of $\{1, 2, 3, \ldots, l\}$ satisfying

1. for all $1 \leq i \leq N$, $|S_i| = a$
2. for all $1 \leq i < j \leq N$, $|S_i \cap S_j| \leq \log N$

Moreover, these sets can be constructed in time polynomial in b and 2^l. We identify the set $\{1, 2, 3, \ldots, l\}$ with the set of new variables $\overline{Y} = \{Y_1, Y_2, \ldots, Y_l\}$. Before we proceed further, we need some notation. We will pick $\delta = (1 - \mu')/2$ to be a non-negative constant. Given, a, μ', δ, we define $\gamma = \frac{2(\mu'+\delta)+1}{1-(\mu'+\delta)}$. Then, we define q to be the smallest prime number between $(a/2)^{\frac{1+\gamma}{2+\gamma}}$ and $2 \cdot (a/2)^{\frac{1+\gamma}{2+\gamma}}$. Also, we set a' to be equal to $(a/2)^{\frac{1}{2+\gamma}}$. Observe that $a/2 \leq a'q \leq a$.

For each i, such that $1 \leq i \leq N$, let $S_i{}'$ be an arbitrary subset of S_i of size equal to $a'q$. For brevity, we rename the sets S_i' as S_i [6]. Let $\rho = (\mu' + \delta)\frac{\log a'q}{\log a'}$ and $D = \frac{\gamma+\rho}{2(1+\gamma)} \cdot a'$.

Often for the ease of notation we will identify the set S_i of $\{1, 2, \ldots, l\}$ with the set of variables $\{Y_j : j \in S_i\}$. We will think of the variables $\{Y_j : j \in S_i\}$ to be arranged in a $a' \times q$ matrix $V(i)$, with the variables placed in the matrix in some order. For every $i \in \{1, 2, 3, \ldots, N\}$, we define $NW_{a',\mu'}(S_i)$ as

$$NW_{a',\mu'}(S_i) = \sum_{\substack{f(z) \in \mathbb{F}_q[z] \\ deg(f) \leq D-1}} \prod_{j \in [a']} V(i)_{jf(j)}.$$

For a point $p = (p_1, p_2, \ldots, p_l) \in \mathbb{F}^l$, we denote by $NW_{a',\mu'}(S_i)|p$, the evaluation of $NW_{a',\mu'}(S_i)$ when the variable Y_j is set to p_j.

Let G be an arbitrary subset of $\mathbb{F}$ of size $Nka' + 1$. We define the hitting set $\mathcal{H}$ as follows.

▶ **Definition 5.7** (Definition of the hitting set $\mathcal{H}$).
$\mathcal{H} = \left\{ (NW_{a',\mu'}(S_1) \,|\, p, NW_{a',\mu'}(S_2) \,|\, p, \ldots, NW_{a',\mu'}(S_N) \,|\, p) : p \in G^l \right\}$.

We now proceed to prove the correctness of the construction. We first prove the following lemma which shows that $\mathcal{H}$ is explicit and has the correct size as per Theorem 5.6.

▶ **Lemma 5.8.** *The set $\mathcal{H}$ as defined in Definition 5.7 has size at most $(Nka' + 1)^l$ and all its elements can be enumerated in time $a^{a'} \cdot (Nka' + 1)^l \cdot N^{O(1)}$.*

[6] We have replaced the family $\{S_1, S_2, \ldots, S_N\}$ by the set family $\{S_1', S_2', \ldots, S_N'\}$ such that for each $i \in [N]$, $S_i' \subseteq S_i$. Observe that the design based properties of the original system continue to hold. The only thing that changes is that the size of S_i' could be smaller than the size of S_i, by at most a factor 2.

Proof. The size of the set $\mathcal{H}$ is equal to $|G|^l = (Nka' + 1)^l$. The set $\mathcal{H}$ can be enumerated by enumerating through the points p in G^l in some natural order (say lexicographic order) and evaluating the tuple $(NW_{a',\mu'}(S_1)|p, NW_{a',\mu'}(S_2)|p, \ldots, NW_{a',\mu'}(S_N)|p)$ at each of these points. For every point p and subset S_i, the polynomial $NW_{a',\mu'}(S_i)$ can be evaluated in time at most $a^{a'} \times \mathrm{Poly}(N)$ from Lemma 4.2. So, the second part of the lemma follows. ◀

Observe that for our choice of parameters, the above bounds on the size and the time of enumeration are bounded by a function which is subexponential in N.

We now show that for every non-zero polynomial P in the class $\mathcal{C}$, as defined in the statement of Theorem 5.6, there exists a point $p \in \mathcal{H}$, such that $P(p)$ is non-zero. We show this in Lemma 5.9 below. That will complete the proof of Theorem 5.6.

5.3 Correctness of the construction

For the rest of this section, we denote N^μ by s.

▶ **Lemma 5.9.** *Let P be a non-zero polynomial in the set $\mathcal{C}$ as defined in the statement of Theorem 5.6, and let $\mathcal{H}$ be the set defined in Definition 5.7. Then, there is a point p in the set $\mathcal{H}$ such that $P(p) \neq 0$.*

Proof. We define

$$P_i(\overline{X}, \overline{Y}) := P(NW_{a',\mu'}(S_1), NW_{a',\mu'}(S_2), \ldots, NW_{a',\mu'}(S_i), X_{i+1}, X_{i+2}, \ldots, X_N)$$

to be the polynomial obtained from P by substituting the variables X_j by $NW_{a',\mu'}(S_j)$, for every $1 \leq j \leq i$.

From the construction of our hitting set, it follows that it would suffice to argue that the polynomial $P_N(\overline{X}, \overline{Y})$ is non-zero. If this was true, then the lemma above will follow from Lemma 5.5, since the degree of any variable $P(\overline{X}, \overline{Y})$ is at most Nka'.

We proceed via contradiction. If possible, let $P_N(\overline{X}, \overline{Y})$ be identically zero. Since $P = P_0(\overline{X}, \overline{Y})$ is non-zero to start with, by a hybrid argument it follows that there is an index i, such that $P_i(\overline{X}, \overline{Y})$ is non-zero while $P_{i+1}(\overline{X}, \overline{Y})$ is identically zero. Observe that P_i is a polynomial in the variables $\overline{Y}$ and $X_{i+1}, X_{i+2}, \ldots, X_N$. In going from P_i to P_{i+1}, we substituted the variable X_{i+1} by the polynomial $NW_{a',\mu'}(S_{i+1})$. Since $P_i(\overline{X}, \overline{Y})$ is non-zero by assumption above, there exists a substitution $\overline{c}$ of all variables apart from $\{Y_j : j \in S_{i+1}\}$ and X_{i+1}, which keeps the polynomial non-zero. Let the polynomial resulting after this substitution be P_i'. From the definitions, it follows that

$$P_i' = P(NW_{a',\mu'}(S_1)|\overline{c}, NW_{a',\mu'}(S_2)|\overline{c}, \ldots, NW_{a',\mu'}(S_i)|\overline{c}, X_{i+1}, X_{i+2}|\overline{c}, \ldots, X_N|\overline{c})\,.$$

Observe that each of the polynomials $NW_{a',\mu'}(S_j)|\overline{c}$ depends only on the variables in the set $S_j \cap S_{i+1}$. From the properties of Nisan-Wigderson designs, and the choice of parameters, the size of this intersection is at most $\log N$. From the definition of P_i and the choice of $\overline{c}$, P_i' is not identically zero. We will think of P_i' as a polynomial in X_{i+1} with the coefficients being polynomials in the variables in the set $\{Y_j : j \in S_{i+1}\}$. Now, we know that the the polynomial P_{i+1}' obtained by substituting X_{i+1} by $NW_{a',\mu'}(S_{i+1})$ is identically zero. Hence, it must be the case that $X_{i+1} - NW_{a',\mu'}(S_{i+1})$ is a factor of P_i'.

To proceed further, we need the following claim.

▶ **Claim 5.10.** *P_i' as defined above can be represented as*

$$P_i' = \sum_{r=1}^{T} \prod_{j=1}^{d} Q_{rj}'$$

such that each of the polynomials Q_{rj}' depends on at most $s \log N$ variables.

Proof. Recall that P can be represented as

$$P = \sum_{i=1}^{T} \prod_{j=1}^{d} Q_{ij}$$

where each Q_{ij} is a polynomial in at most $s = N^{\mu}$ variables. In going from P to P_i', we have substituted each of the variables outside the set $\{Y_j : j \in S_{i+1}\} \cup \{X_{i+1}\}$ by either a constant or by the polynomial $NW_{a',\mu'}(S_j)|\bar{c}$ (which is a polynomial in at most $|S_j \cap S_{i+1}| \leq \log N$ variables) for some j. In either case, after substitution, the polynomials Q'_{rj} obtained from Q_{rj} depends on at most $s \log N$ variables, since Q_{rj} depended on at most s variables. This completes the proof of the claim. ◀

Moreover, since the individual degree of variables in P is at most k, the individual degree of X_{i+1} in P_i' is at most k. The goal now is to invoke Lemma 5.3, which would imply that $NW_{a',\mu'}(S_{i+1})$ also has a small circuit as a sum of product of polynomials in *few* variables, and together with the lower bound from Theorem 4.13, this would lead to a contradiction. We essentially follow this outline. Formally, we use the following claim to complete the proof of Lemma 5.9. We defer the proof of the claim to the end.

▶ **Claim 5.11.** *If $(X_{i+1} - NW_{a',\mu'}(S_{i+1}))$ divides P_i', then $NW_{a',\mu'}(S_{i+1})$ can be written as*

$$NW_{a',\mu'}(S_{i+1}) = \sum_{r=1}^{I'} \prod_{j=1}^{d'} \Gamma_{rj}$$

where

1. $I' \leq (da'^2 + 1) \cdot \binom{k+a'+1}{k+1} \times \binom{T\cdot(k+1)^3+a'}{a'}^{k+1}$,
2. $d' \leq d \cdot a'$, *and*
3. *each Γ_{rj} depends on at most $s \log N$ variables.*

From our choice of parameters, recall that

$$a = N^{\mu/\mu'} \cdot \log^{1/\mu'} N$$

and

$$s = N^{\mu}.$$

Therefore, $s \log N \leq N^{\mu} \cdot \log N \leq a^{\mu'}$. To complete the proof, we observe that by Theorem 4.13, we must have

$$I'd' \geq (a')^{\Omega(\sqrt{a'})}.$$

But, for our choice of parameters,

1. $I' \leq (da'^2 + 1) \cdot \binom{k+a'}{k} \times \binom{T\cdot(k+1)^3+a'}{a'}^{k+1} \leq da^{O(Tk^4)} \leq da'^{O(Tk^4)}$ (since a and a' are polynomially related)
2. $d' \leq da'$

This implies that $I'd' \leq d^2 a^{O(Tk^4)}$. From our choice of parameters, $s \log N < a^{\mu'}$ and $Tk^4 + 2\log d \in o(\sqrt{a'})$. This contradicts that $I'd' \geq (a')^{\Omega(\sqrt{a'})}$. This completes the proof of Lemma 5.9 assuming Claim 5.11. ◀

We now give a proof of Claim 5.11.

Proof of Claim 5.11. From Claim 5.10, we know that

$$P'_i = \sum_{r=1}^{T} \prod_{j=1}^{d} Q'_{rj}$$

such that each Q'_{rj} depends on at most $s \log N$ variables. Since P'_i is not identically zero and $NW_{a',\mu'}(S_{i+1})$ is a root of P'_i, it follows from Lemma 5.2 that there is an integer λ such that $0 \leq \lambda \leq k-1$ and,

$$\frac{\partial^{\lambda} P'_i}{\partial X_{i+1}^{\lambda}}(NW_{a',\mu'}(S_{i+1})) = 0$$

and

$$\frac{\partial^{\lambda+1} P'_i}{\partial X_{i+1}^{\lambda+1}}(NW_{a',\mu'}(S_{i+1})) \neq 0\,.$$

From Lemma 5.1 it follows that $\tilde{P}'_i = \frac{\partial^{\lambda} P'_i}{\partial X_{i+1}^{\lambda}}$ can also be expressed as

$$\tilde{P}'_i = \sum_{r=1}^{T'} \prod_{j=1}^{d} \tilde{Q}_{ij}$$

where $T' \leq T \cdot (k+1)^2$ and each of the $\tilde{Q}_{rj}$ depends on at most $s \log N$ variables.

Observe that, $\tilde{P}'_i$ vanishes when $NW_{a',\mu'}(S_{i+1})$ is substituted for X_{i+1}, while its derivative with respect to X_{i+1} does not vanish identically at $X_{i+1} = NW_{a',\mu'}(S_{i+1})$. So, in particular, there is a substitution of the Y variables where the derivative $\frac{\partial \tilde{P}'_i}{\partial X_{i+1}}$ is nonzero. Since the class of $\Sigma\Pi\,(\Sigma\Pi)^{[s]}$ circuits is closed under translations of variables (from item 2 in Lemma 5.1), we can assume without loss of generality that the derivative is nonzero when all the variables in $\overline{Y}$ are set to zero. Also observe that by this variable translation, we have actually obtained a polynomial $NW'_{a',\mu'}(S_{i+1})$ from $NW_{a',\mu'}(S_{i+1})$. Moreover, the degree of $NW'_{a',\mu'}(S_{i+1})$ is equal to a' and the homogeneous component of degree a' of $NW'_{a',\mu'}(S_{i+1})$ is equal to $NW_{a',\mu'}(S_{i+1})$. Let the polynomial obtained after the variable translation from $\tilde{P}'_i$ as $\tilde{P}''_i$. At this point, the hypothesis of Lemma 5.3 is satisfied by $\tilde{P}''_i$.

Let $\tilde{P}''_i = \sum_{j=0}^{k} C_j(\overline{Y}) \cdot X_{i+1}^j$. Here, $C_j(\overline{Y})$ is a polynomial only in the Y variables and is the coefficient of X_{i+1}^j, when viewing $\tilde{P}''_i$ as an element of $\mathbb{F}[\overline{Y}][X_{i+1}]$. From Lemma 3.4, we know that each of the polynomials C_j can be expressed as a polynomial of the form

$$C_j = \sum_{r=1}^{T_j} \prod_{l=1}^{d} Q''_{rl}$$

where $T_j \leq T' \cdot (k+1) \leq T \cdot (k+1)^3$ and each Q''_{rl} depends on at most $s \log N$ variables.

Hence, by Lemma 5.3, for every $t \geq 0$, there exists a polynomial $R_t \in \mathbb{F}[Z_1, Z_2, \ldots, Z_{k+1}]$ of degree at most t such that

$$\mathsf{Hom}^{\leq t}[NW'_{a',\mu'}(S_{i+1})] = \mathsf{Hom}^{\leq t}[R_t(C_0, C_1, \ldots, C_k)]\,.$$

The goal now is to obtain a representation of $NW_{a',\mu'}(S_{i+1})$ as a sum of products of polynomials in few variables and show that this contradicts the lower bound in Theorem 4.13.

$NW'_{a',\mu'}(S_{i+1})$ is a polynomial of degree at most a'. So, there is a polynomial $R_{a'}$ of degree at most a' in $k+1$ variables such that

$$NW'_{a',\mu'}(S_{i+1}) = \mathsf{Hom}^{\leq a'}[R_{a'}(C_0, C_1, \ldots, C_k)]\,.$$

From the discussion on the relation between $NW'_{a',\mu'}(S_{i+1})$ from $NW_{a',\mu'}(S_{i+1})$, we also know that

$$NW_{a',\mu'}(S_{i+1}) = \mathsf{Hom}^{a'}[NW'_{a',\mu'}(S_{i+1})] = \mathsf{Hom}^{a'}[R_{a'}(C_0, C_1, \ldots, C_k)]\,.$$

Since $R_{a'}$ is a polynomial in $k+1$ variables of degree a', the number of monomials in $R_{a'}$ is at most $\binom{a'+k+1}{k+1}$. Therefore, we can represent $R_{a'}(C_0, C_1, \ldots, C_k)$ as a sum of products of the C_j's, with the sum fan-in at most $\binom{a'+k+1}{k+1}$ and the product fan-in at most a'. Moreover, each of the product gates in this representation takes the polynomials C_j's as inputs. We know that each C_j can be written as

$$C_j = \sum_{r=1}^{T_j} \prod_{l=1}^{d} Q''_{rl}$$

where each Q''_{rl} is a polynomial in at most $s \log N$ variables, and the top sum fan-in T_j is at most $T \cdot (k+1)^3$. For any t, the polynomial C_j^t, has a similar representation with the top sum fan-in at most $\binom{T\cdot(k+1)^3+t}{t}$. Therefore, any product of fan-in at most a' in the C_j's can be written as a sum of product of polynomials in at most $s \log N$ variables, with top fan-in at most

$$\binom{T \cdot (k+1)^3 + a'}{a'}^{k+1}$$

since each C_j is raised to a power of at most a' and there are $k+1$ such C_j's. Therefore, $R_{a'}(C_0, C_1, \ldots, C_k)$ can be written as

$$R_{a'}(C_0, C_1, \ldots, C_k) = \sum_{r=1}^{I} \prod_{j=1}^{d'} \Gamma'_{rj}$$

such that

1. $I \leq \binom{k+a'+1}{k+1} \times \binom{T\cdot(k+1)^3+a'}{a'}^{k+1}$
2. $d' \leq d \cdot a'$
3. Each Γ'_{rj} depends on at most $s \log N$ variables

We would now like to extract the homogeneous part of degree a' of $R_{a'}(C_0, C_1, \ldots, C_k)$, which we know is equal to $NW_{a',\mu'}(S_{i+1})$. We do this by a standard application of Lemma 3.5. Since we are interested only in the homogeneous part of degree a', we can assume without loss of generality that each of the polynomials Γ'_{rj} is of degree at most a' (we can discard all monomials of degree larger than a' in each of the Γ'_{rj}, since they do not contribute to the homogeneous component of degree a' of $R_{a'}(C_0, C_1, \ldots, C_k)$). Hence, the degree of $R_{a'}(C_0, C_1, \ldots, C_k)$ is upper bounded by $da' \cdot a'$. So, from Lemma 3.5, we can extract the homogeneous component of degree a' of $R_{a'}(C_0, C_1, \ldots, C_k)$ by blowing up the top fan-in by a factor of at most $da'^2 + 1$. Hence, $NW_{a',\mu'}(S_{i+1})$ can be expressed as

$$NW_{a',\mu'}(S_{i+1}) = \sum_{r=1}^{I'} \prod_{j=1}^{d'} \Gamma_{rj}$$

where

1. $I' \leq (da'^2+1) \cdot \binom{k+a'+1}{k+1} \times \binom{T\cdot(k+1)^3+a'}{a'}^{k+1}$,
2. $d' \leq d \cdot a'$, and
3. each Γ_{rj} depends on at most $s \log N$ variables. ◀

We remark that if the value of s was $\log^{O(1)} N$ to start with, the same proof as above goes through with l and a being set to polynomials of sufficiently high degree in $\log N$. The size of the hitting set and the time to construct it in this case are upper bounded by a quasipolynomial function in N.

6 Open problems

We conclude with some open problems.

1. An intriguing open question is to obtain PIT for $\Sigma\Pi(\Sigma\Pi)^{[s]}$ circuits without the restriction on the individual degree. The strategy in this paper relies on hardness randomness tradeoffs for bounded depth circuits [6]. The tradeoffs in [6] crucially use the fact that the individual degree is bounded.
2. Another related question would be to get any non-trivial PIT (even subexponential) for the sum of constant many products of degree two polynomials.
3. A related question of interest is to obtain non-trivial PIT for sums of products of polynomials in few variables with bounded individual degree but without a restriction on the top fan-in.
4. It would also be interesting to understand if one could obtain any non-trivial PIT for slightly non-multilinear depth four circuits (say individual degree at most 2) with bounded top fan-in. A natural strategy for this question would be to reduce it to the case of $\Sigma\Pi(\Sigma\Pi)^{[s]}$ circuits by either expanding out the polynomials Q_{ij} which depend on too many variables or use a partial derivative like trick, as in [5]. The immediate challenge in this case is that the top fan-in seems to increase by any of these tricks and the calculations in this paper seem to not work out.

Acknowledgements. We would like to thank Rafael Oliveira for many helpful discussions regarding hardness-randomness tradeoffs for bounded depth arithmetic circuits at the early stages of this work. We are thankful to the anonymous reviewers at CCC-2016, whose comments helped improve the presentation in the paper.

References

1 M. Agrawal and V. Vinay. Arithmetic circuits: A chasm at depth four. In *FOCS*, 2008.
2 Manindra Agrawal, Chandan Saha, Ramprasad Saptharishi, and Nitin Saxena. Jacobian hits circuits: hitting-sets, lower bounds for depth-d occur-k formulas & depth-3 transcendence degree-k circuits. In *Proceedings of the 44th ACM symposium on Theory of computing*, pages 599–614, 2012.
3 Manindra Agrawal, Chandan Saha, and Nitin Saxena. Quasi-polynomial hitting-set for set-depth-Δ formulas. In *Proceedings of the Forty-fifth Annual ACM Symposium on Theory of Computing*, STOC'13, pages 321–330, New York, NY, USA, 2013. ACM. doi:10.1145/2488608.2488649.
4 Noga Alon. Combinatorial nullstellensatz. *Combinatorics, Probability and Computing*, 8, 1999.
5 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.

6 Zeev Dvir, Amir Shpilka, and Amir Yehudayoff. Hardness-randomness tradeoffs for bounded depth arithmetic circuits. *SIAM J. Comput.*, 39(4):1279–1293, 2009. doi:10.1137/080735850.

7 Michael Forbes. Deterministic divisibility testing via shifted partial derivatives. In *FOCS*, 2015.

8 Michael A. Forbes and Amir Shpilka. On identity testing of tensors, low-rank recovery and compressed sensing. In *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19–22, 2012*, pages 163–172, 2012.

9 Michael A. Forbes and Amir Shpilka. Quasipolynomial-time identity testing of non-commutative and read-once oblivious algebraic branching programs. *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, 0:243–252, 2013.

10 H. Fournier, N. Limaye, G. Malod, and S. Srinivasan. Lower bounds for depth 4 formulas computing iterated matrix multiplication. In *STOC*, 2014.

11 A. Gupta, P. Kamath, N. Kayal, and R. Saptharishi. Approaching the chasm at depth four. In *CCC*, 2013.

12 Ankit Gupta. Algebraic geometric techniques for depth-4 PIT & sylvester-gallai conjectures for varieties. *Electronic Colloquium on Computational Complexity (ECCC)*, 21:130, 2014. URL: http://eccc.hpi-web.de/report/2014/130.

13 Ankit Gupta, Pritish Kamath, Neeraj Kayal, and Ramprasad Saptharishi. Arithmetic circuits: A chasm at depth three. In *FOCS*, 2013.

14 V. Kabanets and R. Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Computational Complexity*, 13(1-2):1–46, 2004.

15 N. Kayal, N. Limaye, C. Saha, and S. Srinivasan. An exponential lower bound for homogeneous depth four arithmetic formulas. In *FOCS*, 2014.

16 Neeraj Kayal. An exponential lower bound for the sum of powers of bounded degree polynomials. *ECCC*, 19:81, 2012. URL: http://eccc.hpi-web.de/report/2012/081.

17 Neeraj Kayal and Chandan Saha. Lower bounds for depth three arithmetic circuits with small bottom fanin. *Electronic Colloquium on Computational Complexity (ECCC)*, 21:89, 2014. URL: http://eccc.hpi-web.de/report/2014/089.

18 Neeraj Kayal and Chandan Saha. Lower bounds for sums of products of low arity polynomials. *Electronic Colloquium on Computational Complexity (ECCC)*, 2014.

19 Neeraj Kayal, Chandan Saha, and Ramprasad Saptharishi. A super-polynomial lower bound for regular arithmetic formulas. In *STOC*, 2014.

20 P. Koiran. Arithmetic circuits : The chasm at depth four gets wider. *TCS*, 2012.

21 Mrinal Kumar and Shubhangi Saraf. The limits of depth reduction for arithmetic formulas: It's all about the top fan-in. In *STOC*, 2014.

22 Mrinal Kumar and Shubhangi Saraf. On the power of homogeneous depth 4 arithmetic circuits. *FOCS*, 2014.

23 Partha Mukhopadhyay. Depth-4 identity testing and noether's normalization lemma. *Electronic Colloquium on Computational Complexity (ECCC)*, 2015.

24 Noam Nisan. Lower bounds for non-commutative computation (extended abstract). In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing, May 5-8, 1991, New Orleans, Louisiana, USA*, pages 410–418, 1991.

25 Noam Nisan and Avi Wigderson. Hardness vs randomness. *Journal of Computer and System Sciences*, 49(2):149–167, 1994.

26 Noam Nisan and Avi Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994. doi:10.1016/S0022-0000(05)80043-1.

27 Ran Raz. Separation of multilinear circuit and formula size. *Theory of Computing*, 2(1):121–135, 2006. doi:10.4086/toc.2006.v002a006.

28 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.

29 Nitin Saxena. Diagonal circuit identity testing and lower bounds. *Electronic Colloquium on Computational Complexity (ECCC)*, 14(124), 2007.

30 J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *Journal of ACM*, 27(4):701–717, 1980.

31 A. Shpilka and A. Wigderson. Depth-3 arithmetic circuits over fields of characteristic zero. *Computational Complexity*, 10(1):1–27, 2001.

32 Amir Shpilka. Affine projections of symmetric polynomials. In *Proceedings of the 16th Annual Conference on Computational Complexity*, CCC'01, pages 160–, Washington, DC, USA, 2001. IEEE Computer Society.

33 Sébastien Tavenas. Improved bounds for reduction to depth 4 and depth 3. In *MFCS*, pages 813–824, 2013. doi:10.1007/978-3-642-40313-2_71.

34 L. G. Valiant. Completeness classes in algebra. In *STOC*, 1979.

35 Leslie G. Valiant, Sven Skyum, S. Berkowitz, and Charles Rackoff. Fast parallel computation of polynomials using few processors. *SIAM Journal of Computation*, 12(4):641–644, 1983. doi:10.1137/0212043.

36 R. Zippel. Probabilistic algorithms for sparse polynomials. *Symbolic and algebraic computation*, pages 216–226, 1979.

A Calculations

$$Td^2n^3 \cdot rs \geq \frac{\frac{1}{n^{O(1)}}\min\left(\frac{p^r}{4^r}\cdot\binom{N}{r}\cdot\binom{N}{m},\binom{N}{m+n-r}\right)}{2^{O(\sqrt{n})}\cdot\binom{N}{m+rs}\cdot\binom{n+r}{r}}.$$

We first estimate the ratio $\frac{\binom{N}{m+n-r}}{\binom{N}{m+rs}\cdot\binom{n+r}{r}}$:

$$\frac{\binom{N}{m+n-r}}{\binom{N}{m+rs}\cdot\binom{n+r}{r}} \geq \frac{(m+rs)!}{(m+n-r)!}\frac{(N-m-rs)!}{(N-m-(n-r))!}\cdot\left(\frac{r}{e(n+r)}\right)^r$$

Here we use the fact that $\binom{n+r}{r} \leq \left(\frac{e(n+r)}{r}\right)^r$. Now, approximating the ratios using Lemma 3.3 and substituting $m = \frac{N}{2}(1 - r\frac{\ln n}{n})$, we get

$$\begin{aligned}\frac{\binom{N}{m+n-r}}{\binom{N}{m+rs}\cdot\binom{n+r}{r}} &\geq \left(\frac{N-m}{m}\right)^{n-r-rs}\cdot\left(\frac{r}{e(n+r)}\right)^r \\ &\geq \exp\left(\frac{r\ln n}{n}\cdot(n-r-rs) - r\ln\frac{e(n+r)}{r}\right)\end{aligned}$$

Since $r = \Theta(\sqrt{n})$, we get that the ratio is at least $\exp\left(r\ln n((n-r-rs)/n - \frac{1}{2} + o(1))\right)$, which is $\exp(\Omega(\sqrt{n}\ln n))$.

Next we estimate the ratio $\frac{\left(\frac{p^r}{4^r}\cdot\binom{N}{r}\cdot\binom{N}{m}\right)}{\binom{N}{m+rs}\cdot\binom{n+r}{r}}$:

$$\begin{aligned}
\frac{\left(\frac{p^r}{4^r}\cdot\binom{N}{r}\cdot\binom{N}{m}\right)}{\binom{N}{m+rs}\cdot\binom{n+r}{r}} &\geq \frac{p^r}{4^r}\cdot\frac{(m+rs)!}{m!}\cdot\frac{(N-m-rs)!}{(N-m)!}\cdot\frac{N!}{(N-r)!}\cdot\frac{n!}{(n+r)!}\\
&\geq \frac{p^r}{4^r}\cdot\left(\frac{m}{N-m}\right)^{rs}\cdot\left(\frac{N}{n}\right)^r\\
&\geq \frac{p^r}{4^r}\cdot\left(1-2.01r\frac{\ln n}{n}\right)^{rs}\cdot\left(\frac{N}{n}\right)^r\\
&\geq \frac{1}{4^r}\exp\left(-r(\mu+\delta)\ln N-2.01r^2s\frac{\ln n}{n}+r\ln(N/n)\right)
\end{aligned}$$

Here, we used Lemma 3.3 in the second step and substituted $p = N^{-(\delta+\mu)}$ in the last step. Now, substituting $2n^{2+\gamma} \geq N \geq n^{2+\gamma}$, the exponent is at least

$$r\ln n(-(\mu+\delta)(2+\gamma)-2.01rs/n+(1+\gamma))\,.$$

This is at least

$$r\ln n(-(\mu+\delta)(2+\gamma)-2.01rs/n+(1+\gamma))\,.$$

Now, plugging back the value of γ, the exponent is at least $(2-2.01rs/n)r\ln n$. We have chosen rs such that $rs/n < 0.001$. Therefore, the ratio we set out to lower bound is at least $\exp(\Omega(\sqrt{n}\ln n))$.