ON CIRCUIT COMPLEXITY CLASSES AND ITERATED MATRIX MULTIPLICATION

BY FENGMING WANG

A dissertation submitted to the

Graduate School—New Brunswick

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

Graduate Program in Computer Science

Written under the direction of

Eric Allender

and approved by

_____

_____

_____

_____

New Brunswick, New Jersey

January, 2012

ABSTRACT OF THE DISSERTATION

On Circuit Complexity Classes and Iterated Matrix Multiplication

by Fengming Wang

Dissertation Director: Eric Allender

In this thesis, we study small, yet important, circuit complexity classes within NC^1, such as ACC^0 and TC^0. We also investigate the power of a closely related problem called Iterated Matrix Multiplication and its implications in low levels of algebraic complexity theory. More concretely,

- We show that extremely modest-sounding lower bounds for certain problems can lead to non-trivial derandomization results.
  - If the word problem over S_5 requires constant-depth threshold circuits of size $n^{1+\epsilon}$ for some $\epsilon > 0$, then any language accepted by uniform polynomial-size probabilistic threshold circuits can be solved in subexponential time (and more strongly, can be accepted by a uniform family of deterministic constant-depth threshold circuits of subexponential size.)
  - If there are no constant-depth arithmetic circuits of size $n^{1+\epsilon}$ for the problem of multiplying a sequence of n 3-by-3 matrices, then for every constant d, black-box identity testing for depth-d arithmetic circuits with bounded individual degree can be performed in subexponential time (and even by a uniform family of deterministic constant-depth AC circuits of subexponential size).

- ACC_m circuits are circuits consisting of unbounded fan-in AND, OR and MOD_m gates and unary NOT gates, where m is a fixed integer. We show that there exists a language in non-deterministic exponential time which can not be computed by any non-uniform family of ACC_m circuits of quasi-polynomial size and $o(\log \log n)$ depth, where m is an arbitrarily chosen constant.

- We show that there are families of polynomials having small depth-two arithmetic circuits that cannot be expressed by algebraic branching programs of width two. This clarifies the complexity of the problem of computing the product of a sequence of two-by-two matrices, which arises in several settings.

Acknowledgements

I would like to express my gratitude to everybody who helped me during my life at Rutgers.

First and foremost, I am very grateful to my advisor Eric Allender, for his enlightening guidance and unselfish support. Throughout these years he has always been willing and managed to spend time with me, answer my questions and even discuss problems in my personal life. Eric has been very patient to listen to my ideas which were often nonsensical and never tired of my poor English grammar. He is truly a wonderful friend of mine.

I would like to express my gratitude to my co-advisor Mike Saks for his great discussions and for organizing Theory Reading Seminars. Mike has always been full of creative ideas, sharp insights and enthusiastic encouragement. He is a Mathematics genius and an excellent teacher.

I would like to thank Mario Szegedy and Russell Impagliazzo for volunteering to serve on my dissertation committee. I would like to thank Vikraman Arvind, Troy Lee, Nikos Leonardos and Rahul Santhanam for collaborating with me.

Rutgers has a fantastic group of faculty on Theoretical Computer Science and Mathematics. Over the years, I have benefited a lot from courses taught by Jeff Kahn, Michael Kiessling, János Komlós, Mike Saks, Bill Steiger, Mario Szegedy, Endre Szemerédi, Van Vu and Chuck Weibel. I would like to thank them for the mind-opening lectures and for allowing me to ask stupid questions.

My fellow students at Rutgers made my journey as a graduate student much easier. The lunch discussions with Devendra Desai, Luke Friedman, Mangesh Gupte, Nikos Leonardos, Rajat Mittal, Peter Richter, Lei Wang and Yixin Xu were filled with fun. Without their companion, the trips to attend various conferences and workshops would not have been so delightful.

I would like to thank Carol DiFrancesco and other staff at Rutgers for their constant patience to solve all of administrative problems. Whenever I needed Carol's favor, she never hesitated

to help me out. To me, Carol is the role model for every Graduate Secretary.

I would like to thank Aduri Pavan, my Master thesis advisor, for his guidance and support during the initial stage of my research life. I would like to thank Xiaoyang Gu, one of my best friends, for bringing me into the world of Complexity Theory and for helping me solve many technical and non-technical problems over the years.

Research conducted in this thesis was supported in part by NSF grants CCF-0830133, CCF-0832787 and CCF-1064785.

Dedication

Dedicated to my parents and my wife Cui, without whose support over the years, this would never have happened.

Table of Contents

Chapter 1

Introduction

In complexity theory, a computational problem is usually represented by a subset of $\{0,1\}^*$, which is called *a language*. Within this research area, one of the major goals is to classify languages according to the amount of resources that are required to decide them. One of the best known models in the literature is the Turing machine and people usually measure the running time and space that are required for various computational problems. In this work, we study another formalism of computation - the circuit model. It treats inputs of different lengths separately, which is similar to the task of circuit design in practice.

A standard Boolean circuit C is a directed acyclic graph with a single sink as the output node, where the source nodes (or leaf nodes) are associated with either input variables or elements in $\{0,1\}$, and the internal nodes are labeled by operations in the Boolean algebra which we call AND, OR and NOT gates. The *size* of C, denoted as SIZE(C), is the cardinality of the set of edges that C contains and the *depth* of C, denoted as DEP(C), is the length of its longest path. Let n be the number of distinct input variables in C; then C computes a Boolean function $f : \{0,1\}^n \to \{0,1\}$ in the obvious manner, namely, by evaluating the Boolean operations in C successively on corresponding inputs. We say a language L is accepted by a family of circuits $\mathcal{C} = \{C_n \mid n \in \mathbb{N}\}$ if for all $n \in \mathbb{N}$, C_n computes L_n where L_n is the Boolean function over the domain $\{0,1\}^n$ which agrees with the characteristic sequence of L. Furthermore, if there exist integer-valued functions s and d such that for all $n \in \mathbb{N}$, SIZE$(C_n) \leq s(n)$ and DEP$(C_n) \leq d(n)$, then we say that L is computable by circuits of size s and depth d.

It is not hard to see that if SAT does not have polynomial-size Boolean circuits, then P $\neq$ NP. Furthermore, the famous Karp-Lipton Theorem [KL80] states that if the polynomial-time hierarchy does not collapse to the second level, which is a widely-believed conjecture in complexity theory, then SAT does not have polynomial-size Boolean circuits. This helped both

motivating great interest in proving circuit lower bounds within the theoretical computer science community, and stimulating the systematic study of the circuit complexity of various computational problems. Although Boolean circuits appear easier to analyze than Turing machines, only moderate success has been achieved in dealing with the most general form of the circuit model. Hence, as many other branches of mathematics did when facing challenging problems, researchers started looking at different restricted versions of the model and hoped that the insights gained in these models could help them bootstrap their knowledge towards how Boolean circuits function in full generality. Among the routes that people pursued, one important direction is to consider circuit families of small depth. It is obvious that every Boolean function over $\{0,1\}^n$ has a depth-two circuit of size at most $O(n2^n)$, which is basically a DNF formula. Hence, in order to have a meaningful theory of circuit complexity, we confine our attention to circuit families of sub-exponential size. Within this regime, many important circuit complexity classes have been considered and in this thesis, we will focus on several small classes inside NC^1 and one related important problem in the arithmetic setting. For detailed coverage and a broader overview on circuit complexity, we refer readers to the textbook by Vollmer [Vol99]. We assume that readers are familiar with the notations for traditional complexity classes, such as $\mathrm{P}, \mathrm{NP}, \mathrm{NEXP}$ and etc. (See [AB09] for the standard treatment.)

1.1 AC^0, $\mathrm{AC}^0(\mathrm{MOD}_p)$ and ACC^0

AC^0 is the class of languages recognized by circuit families of polynomial size and constant depth with unbounded fan-in AND and OR gates and unary NOT gates, where the superscript 0 means constant depth. (Later on, we will use superscript 1 to denote $O(\log n)$ depth.) It is a classic result in many textbooks (For instance, see [Vol99]) on the design of digital circuits that the addition of two n-bit integers can be realized by quadratic-size circuits of depth four, and hence, the corresponding decision problem is in AC^0. This circuit complexity class captures constant-time computation on parallel computers, and coincides with the languages that can be specified exactly in the framework of first-order logic [Imm89, BIS90]. Furthermore, it has a natural connection to the polynomial-time hierarchy with oracles [FSS84, Cai89]. In

the early eighties, researchers demonstrated that the computational power of AC^0 is very limited. Furst, Saxe and Sipser [FSS84] as well as Ajtai [Ajt83] independently utilized the technique of random restrictions to show that for any fixed depth, the PARITY function requires super-polynomial-size AC^0 circuits. Yao [Yao85] refined their technique and improved the lower bound to exponential size. Finally, building on his powerful Switching Lemma, Håstad [Hås86] proved the optimal trade-off between size and depth in terms of AC^0 circuits for the PARITY function. Roughly speaking, the Switching Lemma says that with high probability, the sub-functions obtained under random restrictions have constant-size decision tree complexity. Razborov [Raz95] presented another interesting proof of this statement. Beame [Bea] showed that further extensions of the Switching Lemma have fruitful consequences in proving lower bounds for the length of propositional proofs. In fact, the Switching Lemma not only provides deterministic lower bounds, but also states that the PARITY function remains very hard to approximate by AC^0 circuits. This enabled Nisan and Wigderson [NW94] to construct pseudo-random generators to de-randomize probabilistic AC^0 circuits.

Definition 1 *Probabilistic circuits take an input divided into two pieces, the actual input and the random input. We say an input x is accepted by such a circuit C if, with respect to the uniform distribution U_R over all possible random inputs, $Pr_{r\sim U_R}[C(x,r)=1] \geq \frac{2}{3}$ while x is rejected by C if $Pr_{r\sim U_R}[C(x,r)=1] \leq \frac{1}{3}$.*

The above achievements turned researchers' attention to a slightly generalized model, namely, AC^0 circuits equipped with unbounded fan-in PARITY gates. The class of languages accepted by such families of circuits is denoted by $AC^0(\oplus)$ or $AC^0(\text{MOD}_2)$, where for any integer m and any finite string $x = x_0x_1...x_{|x|-1}$, $\text{MOD}_m(x)$ evaluates to 1 if $\Sigma_{i=0}^{|x|-1} x_i = 0 \pmod m$, and 0 otherwise. Razborov [Raz87] showed that $AC^0(\text{MOD}_2)$ circuits are unable to compute the MAJORITY function. Smolensky [Smo87] extended Razborov's method to prove that if p is a prime and q is not a power of p, then the MOD_q function is not computable by any family of constant-depth $AC(\text{MOD}_p)$ circuits of size 2^{n^δ} for some $\delta > 0$, where $AC(\text{MOD}_p)$ circuits are Boolean circuits consisting of unbounded fan-in AND, OR and MOD_p gates and unary NOT gates. These conclusions imply that the MAJORITY function is outside the class $AC^0(\text{MOD}_p)$ as well as many other problems such as Multiplication and Transitive Closure [CSV84].

Given the previous success of applying the algebraic method for establishing lower bounds in terms of AC^0 and $AC^0(MOD_p)$ circuits, it is natural to wonder: what happens if one considers a fixed set of arbitrary moduli, or equivalently, substitutes the prime modulus by a composite modulus? This leads us to the complexity class ACC^0, which stands for "AC^0 with counters", where the circuit families are allowed to have various MOD gates for a constant number of moduli. The definition of ACC^0 was implicit in the work of Barrington [Bar89]. Building on the techniques developed by Toda [Tod89], Yao [Yao90] and Beigel and Tarui [BT94] were able to provide nontrivial upper bounds for ACC^0. They showed that Boolean functions recognized by ACC^0 circuits can be computed by depth-two circuits of a very special form, called SYM^+ circuits, where the top gate is a symmetric gate of quasi-polynomial-size fan-in and the bottom gates are AND gates, each of which is connected to at most polylogarithmic input variables. This phenomenon of depth reduction was previously identified for AC^0 and $AC^0(MOD_p)$ circuits as well [All89, AH94, KVVY93, Tar93, ABFR94].

Definition 2 *Let $\mathcal{C} = (C_n)_{n\in\mathbb{N}}$ be a circuit family. The direct connection language L_{DC} of $\mathcal{C}$ is the set of all tuples having either the form $\langle n, p, q, b\rangle$ or $\langle n, p, d\rangle$, where*

- *If $q = \epsilon$, then b is the type of gate p in C_n;*
- *If q is the binary encoding of k, then b is the kth input to p in C_n.*
- *The gate p has fan-in d in C_n.*

The circuit family $\mathcal{C}$ is DLOGTIME*-uniform if there is a deterministic Turing machine that accepts L_{DC} in linear time.*

Allender and Gore [AG94] observed that the aforementioned transformation can be performed in a DLOGTIME-uniform manner and they combined it with other techniques to prove that the PERMANENT function is not computable by DLOGTIME-uniform ACC^0 circuit families of sub-exponential size.

People also found various characterizations for languages in ACC^0. Barrington and Thérien [BT88] showed that ACC^0 is captured exactly by bounded-width programs of polynomial size over solvable monoids, which confirmed that ACC^0 is the most natural subclass of NC^1. (See

Section 1.2 or [Bar89].) Hansen [Han06] showed that logspace-uniform planar circuits of constant width and polynomial size compute precisely logspace-uniform ACC^0. Allender, Datta and Roy [ADR05] proved that ACC^0 is precisely what can be computed with constant-width circuits of polynomial size and polylogarithmic genus. Hansen and Koucký [HK09] showed that ACC^0 can be computed by probabilistic polynomial-size CC^0 circuits with $O(\log n)$ many random bits, where CC^0 circuits are constant-depth circuits solely consisting of modular counting gates. They also proved that polynomial-size planar branching programs correspond exactly to ACC^0.

Given the above machinery, it looked very likely that super-polynomial lower bounds against ACC^0 circuits for natural problems, say SAT, would not be very far from reach, since researchers had built up seemingly more than enough tools to nail down their computational power. However, for more than two decades, people could not even answer a much simpler question, namely, whether $\mathrm{NEXP} \not\subseteq \mathrm{ACC}^0$ or not. The latter problem had become well-known as one of the major challenges in theoretical computer science until it was recently solved by Williams [Wil11]. In [Wil10], Williams proposed a research program which tried to establish circuit lower bounds via designing fast satisfiability algorithms for Circuit-SAT problems. The next year, Williams [Wil11] succeeded in carrying out the program by proving an ingenious super-polynomial lower bound for NEXP against non-uniform constant-depth ACC circuits of polynomial size, thereby answering this notorious long-standing open question affirmatively.

In Chapter 3, we extend Williams' lower bound result [Wil11] to the non-constant-depth setting and show that

Theorem 3 *There is a language in* NEXP *that does not have non-uniform* ACC *circuits of quasi-polynomial size and* $o(\log \log n)$ *depth.*

In relation to Theorem 3, we would like to mention a few examples of earlier work giving super-polynomial size bounds for AC and $\mathrm{AC}(\mathrm{MOD}_p)$ circuits of non-constant depth. Håstad [Hås86] proved that the PARITY function can not be computed by families of AC circuits of polynomial size and depth at most $\frac{c \log n}{\log \log n}$ for some positive constant c. This result found many applications in proving lower bounds for the parallel random access machine model (PRAM), which is one of the widely adopted models of parallel computation. For instance, Beame and

Håstad [BH89] exhibited the optimal $\Omega(\frac{\log n}{\log \log n})$ lower bounds on the time for CRCW (Concurrent Read and Concurrent Write) PRAM with polynomially many processors to compute the PARITY function and related problems. The technique developed by Razborov [Raz87] and Smolensky [Smo87] works in the regime of non-constant-depth circuit lower bounds. More precisely, one can adapt their polynomial method to show that the same MOD_q function remains hard even for $\text{AC}(\text{MOD}_p)$ circuits of polynomial size and $o(\frac{\log n}{\log \log n})$ depth. Even though Theorem 3 is exponentially worse in terms of circuit depth, note that it holds for the more powerful ACC circuit model.

1.2 TC^0 and NC^1

TC^0 is the class of languages computable by constant-depth circuit families of polynomial size with unary NOT gates and threshold gates of unbounded fan-in. TC^0 contains ACC^0 and many important problems are complete for TC^0, for instance, multiplication and division of two n-bit integers and sorting n n-bit numbers [CSV84, HAB02]. TC^0 provides the complexity theoretic characterization for computation by neural networks [PS88, PS89]. Shaltiel and Viola [SV10] showed that it is the smallest complexity class where one can perform *hardness amplification*, which is a very useful procedure in both theory and practice. Furthermore, there exists a natural correspondence between TC^0 and the counting hierarchy [Wag86]. (See also [AW93].) Allender et al. [ABKPM09] exploited this fact and showed that many interesting problems in numerical analysis can be solved within the counting hierarchy, which somewhat improved the previous PSPACE upper bounds.

NC^1 is the class of languages solvable by circuit families of polynomial size and $O(\log n)$ depth which consist of AND, OR and NOT gates of fan-in at most two. Boolean formula evaluation can be achieved in NC^1 [Bus93] and deciding every regular set is in NC^1 [Bar89]. In the mid eighties, Barrington [Bar89] unveiled a surprising relationship between NC^1 circuits and bounded-width branching programs. More formally, he proved that the computation of group programs over any *non-solvable* group captures exactly the languages in NC^1.

It is not hard to see that TC^0 is a subclass of NC^1. Yao [Yao89] conjectured that this containment is in fact strict; however, even the question whether NEXP $\subseteq$ TC^0 or not is still

far from the reach of current techniques. In the following paragraph, we survey a subset of existing lower bounds against TC^0 circuits. (For detailed treatment, see [Raz92] and [All96].)

In the uniform setting, Allender [All99] showed that the PERMANENT function is not computable by DLOGTIME-uniform polynomial-size TC^0 circuit families. Koiran and Perifel [KP09] extended this result and showed that the PERMANENT function requires DLOGTIME-uniform threshold circuits of super-polynomial size or $\Omega(\log \log n)$ depth. Much less is known for non-uniform TC^0 circuits. Hajnal et al. [HMP$^+$93] proved that TC^0 circuits of depth three are exponentially more powerful than depth-two TC^0 circuits. In [She07], Sherstov exhibited a function in AC^0 which requires exponential-size TC^0 circuits of depth two, via lower bounds in communication complexity. Researchers also considered various restricted versions of depth-three TC^0 circuits [KP94, HG91, RW93, Gro94, HM04] and Sitharam [Sit96] showed how one could obtain some of these lower bounds under a unified framework. For general depth d, Impagliazzo, Paturi and Saks [IPS97] proved that there exist universal constants $c > 0$ and $\theta \leq 3$ such that any TC^0 circuits of depth d which compute the PARITY function on n variables must be of size at least $n^{1+c\theta^{-d}}$. Note that the quality of this lower bound diminishes as d grows and currently there are no super-linear lower bounds against general TC^0 circuit families that are independent of d.

In the mid nineties, Razborov and Rudich [RR97] identified a significant obstacle to futher progress of circuit lower bounds. They defined the notion of "Natural Proof" and showed that most of the known techniques for circuit lower bounds were natural. More importantly, they proved that if pseudo-random function generators can be computed in TC^0, then there are no natural proofs for super-polynomial-size lower bounds against TC^0 circuits. Since Naor and Reingold [NR04] showed that such cryptographically secure functions do exist in TC^0 under reasonable assumptions such as the hardness of factoring integers, the future development of circuit lower bounds requires that the new proof strategies must be unnatural. Recently two approaches have been proposed in the effort to circumvent the natural proof barrier. Williams [Wil10] proved that if fast co-nondeterministic algorithms for the satisfiability of TC^0 circuits can be obtained, then NEXP $\not\subseteq$ TC^0. It does not seem that this falls under the framework of natural proofs, since Williams' argument relies heavily on diagonalization which is inherently unnatural. The other route was initiated by Allender and Koucký [AK10]. They suggested

taking advantage of extra properties of problems in NC^1. One such example is the word problem for S_5, denoted as WP, where S_5 is the permutation group over five elements. Barrington [Bar89] showed that it is a complete problem for NC^1. Allender and Koucký observed that WP possesses the nice property of being *strongly downward-self-reducible*. (This notion was first discovered by Goldwasser et al. [GGH$^+$07].) In order to formulate the definition of this notion, it is necessary to introduce oracle circuits. Fix the alphabet to be Σ.

Definition 4 *Let n, m be two positive integers and $f_{n,m} : \Sigma^n \to \Sigma^m$ be an arbitrary function. An oracle gate $G^{f_{n,m}}$ with respect to $f_{n,m}$ is a gate which takes $x \in \Sigma^n$ as input and produces $f(x) \in \Sigma^m$ as output.*

Let $\mathcal{F} = \{f_{n,m} \mid n, m \in \mathbb{N}\}$ be a family of functions. Oracle circuits with respect to $\mathcal{F}$ *are standard circuits enhanced by the family of oracle gates $\{G^{f_{n,m}} \mid f_{n,m} \in \mathcal{F}\}$. Note that the description of oracle circuits is not dependent on the functionality of $\mathcal{F}$, but the outputs they compute do rely on the implementation of the underlying oracle.*

Definition 5 ([AK10]) *Let m be a fixed integer and $\mathcal{F} = \{f_n : \Sigma^n \to \Sigma^m \mid n \in \mathbb{N}\}$ be a family of functions. Let $\mathcal{C} = \{C_n \mid n \in \mathbb{N}\}$ be a family of oracle circuits with respect to $\mathcal{F}$ which is of size at most $s(n)$, and let $l(n) : \mathbb{N} \to \mathbb{N}$ be a function such that $\forall n,\ l(n) < n$. We say that f_n is* downward-reducible *to $f_{l(n)}$ via $\mathcal{C}$ if C_n computes f_n and moreover, C_n only contains oracle gates from $\{G^{f_p} \mid p \leq l(n)\}$. We say $\mathcal{F}$ is* strongly downward-self-reducible via reductions of size $s(n)$ *if for all $0 < \epsilon < 1$, there exists a family of oracle circuits $\mathcal{C}$ of size at most $s(n)$ such that for all sufficiently large $n \in \mathbb{N}$, f_n is downward-reducible to f_{n^ϵ} via $\mathcal{C}$.*

Allender and Koucký proved that WP is strongly downward-self-reducible via linear-size reductions of constant-depth, and building on this result, they were able to amplify any worst-case lower bound for WP. More concretely, they showed that if there exists $\epsilon > 0$ such that WP does not have TC^0 circuits of size $n^{1+\epsilon}$, then WP requires super-polynomial-size TC^0 circuits to compute. They also pointed out that many other problems in NC^1 share the same property, for instance, Boolean formula evaluation and iterated matrix multiplication for matrices of constant dimension.

In Chapter 2, we study another important property of WP, that is, *random-self-reducibility*. This notion has played an important role in many areas of theoretical computer science, for

instance, cryptographical designs, program checking and probabilistically checkable proofs. (See [FF91] and [All10] for more details.) The following definition originated in the work of Feigenbaum and Fortnow [FF91].

Definition 6 ([FF91]) *Let $f : \Sigma^* \to \Sigma^*$ be a function. We say f is* non-adaptively k-random-self-reducible *if there exist polynomial-time computable functions α and β and a polynomial $\gamma : \mathbb{N} \to \mathbb{N}$ such that*

1. *$\forall n \in \mathbb{N}$ and $\forall x \in \Sigma^n$, $f(x) = \alpha(x, r, f(\beta(1, x, r)), ..., f(\beta(k, x, r)))$ with probability at least $\frac{2}{3}$ if r is chosen uniformly at random from $\Sigma^{\gamma(n)}$, and*
2. *$\forall n \in \mathbb{N}$, $\forall x_1, x_2 \in \Sigma^n$ and $\forall 1 \leq i \leq k$, $\beta(i, x_1, r)$ and $\beta(i, x_2, r)$ are identically distributed if r is chosen uniformly at random from $\Sigma^{\gamma(n)}$.*

We show that there exists a constant $k > 0$ such that WP is non-adaptively k-random-self-reducible and furthermore, both α and β can be computed by DLOGTIME-uniform TC^0 circuits. This implies that WP has a very efficient worst-case-to-average-case reduction. Combining the aforementioned strong downward-self-reducibility of WP, we prove that

Theorem 7 *If there exists $\epsilon > 0$ such that* WP *does not have $n^{1+\epsilon}$-size* TC^0 *circuits, then probabilistic* DLOGTIME*-uniform* TC^0 *circuits can be simulated by deterministic* DLOGTIME*-uniform* TC^0 *circuits of sub-exponential size.*

Note that probabilistic non-uniform TC^0 circuits can be derandomized easily via the standard technique [Adl78]. However, this approach relies on existential arguments and is inherently non-uniform. For more background on hardness-based derandomization, see [Kab02].

1.3 On Iterated Matrix Multiplication

In this section, we switch gears and study a closely related computational model - arithmetic circuits. Let the underlying field be $\mathbb{F}$. In this setting, we label the internal nodes of circuits by arithmetic operations (such as $+, -, \times$) instead of Boolean operations. The division operation can be omitted in this context. The constants associated with leaf nodes are arbitrary field

elements, not just $\{0, 1\}$. Therefore, the outputs of arithmetic circuits are formal polynomials over the input variables.

Researchers started the investigation of arithmetic circuits a long time ago. In particular, Valiant [Val79b] formulated an analogue of the P $\stackrel{?}{=}$ NP question purely in terms of polynomials computable by arithmetic circuits of polynomial size. This framework and its extensions were surveyed nicely by several experts [vzG87, Bür00, All04]. Arithmetic circuits also correspond naturally to counting complexity classes [AAB$^+$99, All04]. In this thesis, we will focus on the family of Iterated Matrix Multiplication polynomials, which arises in many settings and plays a central role in algebraic complexity theory. The n^{th} Iterated Matrix Multiplication polynomial of degree d, denoted IMM$_{d,n}$, is the multi-linear polynomial with d^2n variables that is the result of multiplying n d-by-d matrices of indeterminants. We will discuss its connection to the Identity Testing problem for constant-depth arithmetic circuits in Section 1.3.1 and mention an impossibility result for IMM$_{2,n}$ in Section 1.3.2.

1.3.1 Identity Testing for Constant-depth Arithmetic Circuits

The Identity Testing problem for an arithmetic circuit C is to decide whether C computes the zero polynomial. By the Schwartz-Zippel Lemma [Sch80, Zip79], the Identity Testing problem can be solved in randomized polynomial time if $\mathbb{F}$ is sufficiently large. However, it remains as one of the well-known open problems in complexity theory whether the sampling procedure in the Schwartz-Zippel Lemma can be performed deterministically.

About a decade ago, in their seminal work [KI04], Kabanets and Impagliazzo pointed out the close relationship between circuit lower bounds and the identity testing problem for arithmetic circuits. Roughly speaking, they proved that if there exist deterministic polynomial-time algorithms for the Identity Testing problem, then super-polynomial circuit lower bounds can be obtained for either NEXP or the PERMANENT function. Building on a previous result by Allender et al. [AJMV98], Agrawal and Vinay [AV08] showed that black-box derandomization of Identity Testing for depth-four circuits with multiplication gates of small fan-in leads to an almost complete derandomization of Identity Testing for general circuits. These discoveries encouraged a great deal of recent development towards solving the Identity Testing problem for constant-depth arithmetic circuits with unbounded fan-in addition and multiplication gates.

Klivans and Spielman [KS01] proposed to study the Identity Testing problem for depth-three arithmetic circuits of a special form, where the fan-in of the top addition gate is bounded. Hence, these circuits compute polynomials which are the sum of a constant number of multiplication terms, where each multiplication term is a product of linear forms. By bounding the rank of the linear forms appearing in an identity circuit, Dvir and Shpilka [DS05] provided the first deterministic quasi-polynomial-time algorithm for deciding the Identity Testing problem for this type of circuits. After a dramatic series of papers [KS06, KS08, SS09, KS09, SS10], Saxena and Seshadhri [SS11] finally solved the question posed by [KS01] and proved that if $\mathbb{F}$ is sufficiently large, then there is a deterministic Identity Testing algorithm for such circuits which runs in time poly(nd^k), where n is the number of variables, d is the degree of each multiplication term and k is the top fan-in. Notice that the running time of the above algorithms increases exponentially in the top fan-in. Karnin et al. [KMSV10] obtained a sub-exponential-time algorithm for a small class of depth-four multi-linear circuits, however, the running time of their algorithm grows exponentially in the top fan-in as well. Therefore, the Identity Testing problem for general depth-four circuits is still wide open, even with the restriction that the circuits compute multi-linear polynomials.

In counterpoint to these results on unconditional Identity Testing, Kabanets and Impagliazzo [KI04] showed that the framework of hardness-randomness tradeoffs, which originated in Boolean complexity, also applies to arithmetic circuits. Dvir, Shpilka and Yehudayoff [DSY09] introduced this paradigm to the setting of constant-depth arithmetic circuits and they proved that super-polynomial lower bounds for bounded-depth arithmetic circuits imply de-randomization of the Identity Testing problem for arithmetic circuits of bounded depth. In the literature, researchers have successfully obtained some lower bounds for arithmetic circuits [GR00, Shp01, Raz09, RY09], however, these lower bounds can not be directly applied, since either they are too weak to be useful for de-randomizing the Identity Testing problem, or the underlying model is severely restricted such that hardness-based derandomization becomes infeasible. (More details on the current status of arithmetic circuit lower bounds can be found in Raz's paper [Raz10, Section 1.3].) One natural candidate for such hard polynomials is the family of Iterated Matrix Multiplication polynomials $\text{IMM}_{d,n}$ where $d \geq 3$ is a fixed constant. It is a complete problem (under projections) for VNC^1, the arithmetic circuit complexity class

defined by families of polynomials that can be expressed by arithmetic formulae of polynomial size [BOC88, BOC92]. It is obvious that VNC^1 encompasses all families of polynomials computable by arithmetic circuits of bounded depth. We observe that $\text{IMM}_{d,n}$ possesses the property of strong downward-self-reducibility introduced in Section 1.2, thus enabling amplification of any lower bound for $\text{IMM}_{d,n}$ against constant-depth arithmetic circuits. More precisely, we show that

Theorem 8 *If there exists $\epsilon > 0$ such that $\text{IMM}_{d,n}$ does not have constant-depth arithmetic circuits of size $n^{1+\epsilon}$, then $\text{IMM}_{d,n}$ requires super-polynomial-size arithmetic circuits of bounded depth.*

Similar statements were made by Allender and Koucký for the Boolean complexity of $\text{IMM}_{d,n}$ [AK10]. Building on Theorem 8, we prove a derandomization result based on a very mild hardness assumption for $\text{IMM}_{d,n}$.

Theorem 9 *If there are no constant-depth arithmetic circuits of size $n^{1+\epsilon}$ for the polynomial sequence $\text{IMM}_{d,n}$, then for every constant d, black-box Identity Testing for depth-d arithmetic circuits with bounded individual degree can be performed by a uniform family of constant-depth AC^0 circuits of sub-exponential size.*

It is plausible that the slightly super-linear lower bound in the assumption of Theorem 8 might be much easier to achieve, since researchers have obtained some modest lower bound in the general setting. (See [Shp01] for more details.)

1.3.2 Impossibility Result for $\text{IMM}_{2,n}$

Ben-Or and Cleve [BOC88, BOC92] showed that $\text{IMM}_{3,n}$ is complete (under projections) for the class VNC^1, which is the analog of the Boolean class NC^1 in the setting of algebraic complexity. This notation was first formally used by Mahajan and Rao [MR09], and is also sometimes denoted VP_e (corresponding to the subclass of Valiant's class VP of polynomials of polynomial degree that have arithmetic circuits of polynomial size, where we restrict the circuits to be *expressions*). It is natural to wonder if Ben-Or and Cleve's construction is optimal, in terms of dimension. That is: What can one say about $\text{IMM}_{2,n}$?

There are some indications that $\mathrm{IMM}_{2,n}$ should be nearly as powerful as $\mathrm{IMM}_{3,n}$. For instance, Ben-Or and Cleve's completeness argument proceeds by showing that arithmetic formulae can be efficiently evaluated by a restricted type of straight-line program with three registers (and this translates into an implementation with 3-by-3 matrices). In the original conference publication of their results [BOC88], Ben-Or and Cleve credit Coppersmith with the observation that if the underlying ring is commutative and has an element $\frac{1}{2}$ such that $\frac{1}{2} + \frac{1}{2} = 1$, then in fact *two* registers suffice to evaluate any arithmetic formula (albeit via straight-line programs that do not immediately lend themselves to implementation as $\mathrm{IMM}_{2,n}$ computations).

Perhaps the first study of the complexity of $\mathrm{IMM}_{2,n}$ arose in the work of Lipton and Zalcstein [LZ77], who (in modern terminology) showed that the word problem over the free group with two generators (also known as the two-sided Dyck language) is AC^0-reducible to the problem of determining if a product of n two-by-two integer matrices evaluates to the identity matrix. Since the two-sided Dyck language is hard for NC^1 [Rob93], this gives a lower bound on the complexity of evaluating $\mathrm{IMM}_{2,n}$ instances.

This lower bound is rather close to the best known upper bound. The problem of evaluating integer instances of $\mathrm{IMM}_{3,n}$ is complete for the Boolean complexity class GapNC^1 [CMTV98] (consisting of integer-valued functions that have *arithmetic* circuits of polynomial size and logarithmic depth), and every problem in this latter class has *Boolean* circuits of polynomial size, bounded-fan-in, and depth $O(\log n \log^* n)$ [Jun85]. The closeness of these bounds has led some researchers to wonder whether the classes of functions in NC^1 and GapNC^1 are in fact equal [All04], in which case $\mathrm{IMM}_{2,n}$ and $\mathrm{IMM}_{3,n}$ would be inter-reducible under AC^0 reductions.

The NC^1-hardness of $\mathrm{IMM}_{2,n}$ over the integers holds even for restricted cases of the problem. In [AAB$^+$99], it is asserted that counting paths in planar width-two graphs (a restricted case of $\mathrm{IMM}_{2,n}$ over the integers) is hard for NC^1 under ACC^0 reductions. (Mahajan, Saurabh, and Sreenivasaiah [MSS11] have identified and corrected an error in the proof of this claim in [AAB$^+$99].)

On the other hand, there have also been indications that $\mathrm{IMM}_{2,n}$ should be weaker than $\mathrm{IMM}_{3,n}$. Ben-Or and Cleve [BOC92] pointed out that problems over GF(2) having what they called "LBS" straight-line programs (i.e., restricted straight-line programs which they used as a

tool in presenting their completeness result) that use only two registers translate into permutation branching programs of width three, which Barrington [Bar85] showed require exponential size in order to compute the AND function. However, this does not strictly rule out more general computations over $\text{IMM}_{2,n}$.

The AC^0 reductions from problems in NC^1 to $\text{IMM}_{2,n}$ are not projections, which are the usual type of reductions that are used in studying algebraic complexity classes. To illustrate the difference, consider functions in the class GapAC^0; this class consists of functions computed by polynomial-size constant-depth arithmetic circuits over the integers, where the input variables take only Boolean inputs. $\text{GapAC}^0 \subseteq \text{TC}^0 \subseteq \text{NC}^1$ [AAD00], and hence any bit of any function $f \in \text{GapAC}^0$ can be computed by an AC^0 reduction to the problem of multiplying a sequence of 2-by-2 integer matrices. However, any such function f can also be viewed as a polynomial $f(x_1, \ldots, x_n)$ in its input variables, and the AC^0 reduction does not allow us to obtain f from IMM_{2,n^k} by substituting field elements and the variables $x_1, \ldots, x_n$ for the variables of IMM, even though this is possible for IMM_{3,n^k}. In Chapter 4, we will show that, even for fairly simple functions $f \in \text{GapAC}^0$, no such reduction is possible – even if we allow projections to arbitrarily large IMM instances, and even if we greatly enlarge the type of substitutions that considered (beyond the projections that are usually considered in the framework of Valiant's complexity classes).

If we expand the notion of projection, to allow not only variables and field elements to be plugged in for the variables of a polynomial, but also allow variables of IMM instances to be replaced by arbitrary linear expressions, then we obtain an alternative characterization of algebraic branching programs, which were introduced by Nisan in order to study the complexity of determinant and permanent computations in various settings [Nis91].

Definition 10 *An* Algebraic Branching Program *over some field $\mathbb{F}$ and the set of variables $\{x_i \mid 1 \leq i \leq n\}$ is a layered directed acyclic graph with a single source vertex s and exactly one sink vertex t. The layers are numbered as $0, 1, 2, ..., d$; let V_i denote the set of vertices in the ith layer. The source (the sink, respectively) is the unique vertex in V_0 (V_d, respectively). Edges exist only between vertices in adjacent layers (i.e., each edge (a, b) has $a \in V_i$ and $b \in V_{i+1}$ for some $0 \leq i < d$. Each edge e is associated with a linear function l_e over $\mathbb{F}$ in the variables*

$\{x_i \mid 1 \leq i \leq n\}$. *Every directed path* $p = e_1 e_2 ... e_k$ *represents the product* $f_p = \Pi_k^{j=1} l_{e_j}$. *For every vertex* v*, consider the set* $P_{s,v}$ *of all paths from* s *to* v*, the polynomial represented by* v*, denoted as* f_v*, is* $\Sigma_{p \in P_{s,v}} f_p$. *The output of the algebraic branching program is* f_t. *The* width *of the program is* $\max_i |V_i|$.

It follows from [BOC92] that polynomial-size algebraic branching programs of width three (or of any constant width $w \geq 3$) characterize exactly the polynomials in VNC^1. Algebraic branching programs of constant width have been studied by several authors; we cite some recent examples [JR09, Jan08]. We show that width three is optimal; the expressive power of width two algebraic branching programs is severely limited.

Theorem 11 $\forall k \geq 8$, *the polynomial* $f(\overline{x}) = \Sigma_{i=1}^{k} x_{4i-3} x_{4i-2} x_{4i-1} x_{4i}$ *can not be computed by algebraic branching programs of width two over any field* $\mathbb{F}$.

Chapter 2

Uniform Derandomization from Pathetic Lower Bounds

2.1 Background on Derandomization

Hardness-based derandomization is one of the success stories of the past quarter century. The main thread of this line of research dates back to the work of Shamir, Yao, and Blum and Micali [Sha81, Yao82, BM84], and involves showing that, if given a suitably hard function f, one can construct pseudorandom generators and hitting-set generators. Much of the progress on this front over the years has involved showing how to weaken the hardness assumption on f and still obtain useful derandomizations [BFNW93], [AK97], [IW97], [IW01], [KvM02], [ACR99], [ACR98], [ACRT99], [BF99], [MV05], [GW99], [GVW00], [ISW06], [STV01], [SU05], [Uma03]. In rare instances, it has been possible to obtain *unconditional* derandomizations using this framework; Nisan and Wigderson showed that uniform families of probabilistic AC^0 circuits can be simulated by uniform deterministic AC^0 circuits of size $n^{\log^{O(1)} n}$ [NW94]. More often, the derandomizations that have been obtained are conditional, and rely on the existence of functions f that are hard on average. For certain large complexity classes $\mathcal{C}$ (notably including $\#\mathsf{P}$, PSPACE, and exponential time), various types of random self-reducibility and hardness amplification have been employed to show that such hard-on-average functions f exist in $\mathcal{C}$ if and only if there is some problem in $\mathcal{C}$ that requires large Boolean circuits [BFNW93, IW97].

A more recent thread in the derandomization literature has studied the implications of *arithmetic* circuit lower bounds for derandomization. Kabanets and Impagliazzo showed that, if the Permanent requires large *arithmetic circuits*, then the probabilistic algorithm to test if two arithmetic *formulae* (or more generally, two arithmetic circuits of polynomial degree) are equivalent can be simulated by a quick deterministic algorithm [KI04]. Subsequently, Dvir, Shpilka, and Yehudayoff built on the techniques of Kabanets and Impagliazzo, to show that if one could

present a multilinear polynomial (such as the permanent) that requires depth d arithmetic formulae of size 2^{n^ϵ}, then the probabilistic algorithm to test if two arithmetic circuits of depth $d-5$ are equivalent (where in addition, the variables in these circuits have degree at most $\log^{O(1)} n$) can be derandomized to obtain a $2^{\log^{O(1)} n}$ deterministic algorithm for the problem [DSY09].

In this chapter, we combine these two threads of derandomization with the recent insight that, in some cases, extremely modest-sounding (or even "pathetic") lower bounds can be amplified to obtain superpolynomial bounds [AK10]. In order to carry out this combination, we need to identify and exploit some special properties of certain functions in and near NC^1.

- The word problem over S_5 is one of the standard complete problems for NC^1 [Bar89]. Many of the most familiar complete problems for NC^1 have very efficient *strong downward self-reductions* [AK10]. We show that the word problem over S_5, in addition, is *randomly self-reducible*. (This was observed previously by Goldwasser *et al.* [GGH$^+$07], and the idea goes back at least as long as [Kil88]) This enables us to transform a "pathetic" *worst-case* size lower bound of $n^{1+\epsilon}$ on constant-depth threshold circuits, to a superpolynomial size *average-case* lower bound for this class of circuits. In turn, by making some adjustments to the Nisan-Wigderson generator, this average-case hard function can be used to give uniform subexponential derandomizations of probabilistic TC^0 circuits.

- Iterated Multiplication of n three-by-three matrices is a multilinear polynomial that is complete for arithmetic NC^1 [BOC92]. In the Boolean setting, this function is strongly downward self-reducible via self-reductions computable in TC^0 [AK10]. Here we show that there is a corresponding *arithmetic* self-reduction; this enables us to amplify a lower bound of size $n^{1+\epsilon}$ for constant-depth arithmetic circuits, to obtain a superpolynomial lower bound for constant-depth arithmetic circuits. Then, by building on the approach of Dvir *et al.* [DSY09], we are able to obtain subexponential derandomizations of the identity testing problem for a class of constant-depth arithmetic circuits.

The rest of the chapter is organized as follows: In Section 2 we give the preliminary definitions and notation. In Section 3 we convert a modest worst-case hardness assumption to a strong average-case hardness separation of NC^1 from TC^0. We also present slightly weaker

worst-case-to-average-case reductions for logspace and for the classes GapL and GapNC^1. In Section 4 we use this to give a uniform derandomization of probabilistic TC^0 circuits. Finally, in Section 5 we prove our derandomization of a special case of polynomial identity testing under a modest hardness assumption.

2.2 Preliminaries

For any function $s(n)$, TC$^0(s(n))$ consists of languages that are decided by constant-depth circuit families of size at most $s(n)$ which contain only unbounded fan-in MAJORITY gates as well as unary NOT gates. TC^0 $= \cup_{k\geq 0}$TC$^0(n^k)$. TC$^0(\mathsf{SUBEXP}) = \cap_{\delta\geq 0}TC^0(2^{n^\delta})$. The definitions of AC$^0(s(n))$, AC^0, and AC$^0(\mathsf{SUBEXP})$ are similar, although MAJORITY gates are not allowed, and unbounded fan-in AND and OR gates are used instead.

As is usual in arguments in derandomization based on the hardness of some function f, we require not only that f not have small circuits in order to be considered "hard", but furthermore we require that f needs large circuits at *every* relevant input length. This motivates the following definition.

Definition 12 *Let A be a language, and let D_A be the set $\{n : A \cap \Sigma^n \neq \emptyset\}$. We say that $A \in$ io-TC$^0(s(n))$ if there is an infinite set $I \subseteq D_A$ and a language $B \in$ TC$^0(s(n))$ such that, for all $n \in I, A_n = B_n$ (where, for a language C, we let C_n denote the set of all strings of length n in C). Similarly, we define* io-TC^0 *to be* $\cup_{k\geq 0}$io-TC$^0(n^k)$.

Thus A requires large threshold circuits on *all* relevant input lengths if $A \notin$ io-TC^0. (A peculiarity of this definition is that if A is a *finite* set, or A^n is empty for infinitely many n, then $A \notin$ io-TC^0. This differs starkly from most notions of "io" circuit complexity that have been considered, but it allows us to consider "complex" sets A that are empty on infinitely many input lengths; the alternative would be to consider artificial variants of the "complex" sets that we construct, having strings of every length.)

For any circuit complexity class C, uC is its uniform counterpart, consisting of languages that are accepted by $\mathsf{DLOGTIME}$-uniform circuit families. The term "uniform derandomization" in the title of this chapter refers to the fact that we are presenting uniform circuit families

that compute derandomized algorithms; this should not be confused with doing derandomization based on uniform hardness assumptions.

A particularly important complete language for NC^1 is the word problem WP for S_5, where S_5 is the symmetric group over 5 distinct elements [Bar89]. The input to the word problem is a sequence of permutations from S_5 and it is accepted if and only if the product of the sequence evaluates to the identity permutation. The corresponding *search* problem FWP is required to output the exact result of the iterated multiplication. A closely related *balanced* language is BWP, which stands for Balanced Word Problem.

Definition 13 *The input to* BWP *is a pair* $\langle w_1 w_2 .. w_n, S\rangle$*, where* $\forall i \in [1..n],\ w_i \in S_5$*,* $S \subseteq S_5$ *and* $|S| = 60$*. The pair* $\langle w_1 w_2 .. w_n, S\rangle$ *is in* BWP *if and only if* $\Pi_{i=1}^{n} w_i \in S$*.*

It is easy to verify that BWP is complete for NC^1 as well.

In the following sections, let FWP_n be the sub-problem of FWP where the domain is restricted to inputs of length n and let BWP_n be $\mathrm{BWP} \cap \{\langle \phi, S\rangle \mid \phi \in S_5^n,\ S \subseteq S_5,\ |S| = 60\}$. Note that BWP_n accepts exactly half of the instances in $\{\langle \phi, S\rangle \mid \phi \in S_5^n,\ S \subseteq S_5,\ |S| = 60\}$ since $|S_5| = 120$.

The following simplified version of Chernoff's bound turns out to be useful in our application.

Lemma 14 (Chernoff's bound) *Let* $X_1, .., X_m$ *be i.i.d. 0-1 random variables with* $E[X_i] = p$*. Let* $X = \Sigma_{i=1}^{n} X_i$*. Then for any* $0 < \delta \leq 1$*,*

$$Pr[X < (1-\delta)pm] \leq e^{-\frac{\delta^2 pm}{2}}.$$

2.3 The existence of an average-case hard language

2.3.1 Worst-case to Average-case Reduction for NC^1

In this section, we use random self-reducibility to show that, if $\mathrm{NC}^1 \neq \mathrm{TC}^0$, then there are problems in NC^1 that are hard on average for TC^0. First we recall the definition of hardness on average for decision problems.

Definition 15 *Let U_D denote the uniform distribution over all inputs in a finite domain D. For any Boolean function $f : D \to \{0,1\}$, f is $(1-\epsilon)$-hard for a set of circuits S, if, for every $C \in S$, we have that $Pr_{x \sim U_D}[f(x) = C(x)] < 1 - \epsilon$.*

We will sometimes abuse notation by identifying a set with its characteristic function. For languages to be considered hard on average, we consider only those input lengths where the language contains some strings.

Definition 16 *Let Σ be an alphabet. Consider a language $L = \cup_n L_n$, where $L_n = L \cap \Sigma^n$, and let $D_L = \{n : L_n \neq \emptyset\}$. We say that L is $(1-\epsilon)$-hard for a class of circuit families $\mathcal{C}$ if D_L is an infinite set and, for any circuit family $\{C_n\}$ in $\mathcal{C}$, there exists m_0 such that for all $m \in D_L$ such that $m \geq m_0$, $Pr_{x \in \Sigma^m}[f(x) = C(x)] < 1 - \epsilon$.*

The following theorem shows that if FWP $\notin$ io-TC^0, then BWP is hard on average for TC^0.

Theorem 17 *There exist constants $c, \delta > 0$ and $0 < \epsilon < 1$ such that for any constant $d > 0$, if* FWP$_n$ *is not computable by* TC$^0(\delta n(s(n) + cn))$ *circuits of depth at most $d + c$, then* BWP$_n$ *is $(1-\epsilon)$-hard for* TC^0 *circuits of size $s(n)$ and depth d.*

Proof. Let $\epsilon < \frac{1}{4\binom{120}{60}}$. We prove the contrapositive. Assume there is a circuit C of size $s(n)$ and depth d such that $Pr_x[\text{BWP}_n(x) = C(x)] \geq 1 - \epsilon$. We first present a probabilistic algorithm for FWP$_n$.

Let the input instance for FWP$_n$ be $w_1 w_2 \ldots w_n$. Generate a sequence of $n+1$ random permutations $u_0, u_1, \ldots, u_n$ in S_5 and a random set $S \subseteq S_5$ of size 60. Let ϕ be the sequence $(u_0 \cdot w_1 \cdot u_1)(u_1^{-1} \cdot w_2 \cdot u_2)..(u_{n-1}^{-1} \cdot w_n \cdot u_n)$. Note that ϕ is a completely random sequence in S_5^n.

Let us say that ϕ is a "good" sequence if $\forall S' \subset S_5$ with $|S'| = 60$, $C(\langle \phi, S' \rangle) =$ BWP$_n(\langle \phi, S' \rangle)$.

If we have a "good" sequence ϕ (meaning that for *every* set S' of size 60, C gives the "correct" answer BWP$_n(\phi, S')$ on input (ϕ, S')), then we can easily find the unique value r that is equal to $\Pi_{i=1}^n \phi_i$ where $\phi_i = u_{i-1} w_i u_i$, as follows:

- If $C(\phi, S) = 1$, then it must be the case that $r \in S$. Pick any element $r' \in S_5 \setminus S$ and observe that r is the only element such that $C(\phi, (S \setminus \{r\}) \cup \{r'\}) = 0$.

- If $C(\phi, S) = 0$, then it must be the case that $r \notin S$. Pick any element $r' \in S$ and observe that r is the only element such that $C(\phi, (S \setminus \{r'\}) \cup \{r\}) = 1$.

Thus the correct value r can be found by trying all such r'. Hence, if ϕ is good, we have

$$r = \Pi_{i=1}^{n} \phi_i = u_0 w_1 u_1 \Pi_{i=2}^{n} u_{i-1}^{-1} w_i u_i.$$

Produce as output the value $u_0^{-1} r u_n^{-1} = \Pi_{i=1}^{n} w_i = \text{FWP}_n(w)$.

Since $\epsilon < \frac{1}{4\binom{120}{60}}$, a standard averaging argument shows that at least $\frac{3}{4}$ of the sequences in S_5^n are good. Thus with probability at least $\frac{3}{4}$, the probabilistic algorithm computes FWP_n correctly. The algorithm can be computed by a threshold circuit of depth $d + O(1)$ since the subroutines related to C can be invoked in parallel and moreover, the preparation of ϕ and the aggregation of results of subroutines can be done by constant-depth threshold circuits. Its size is at most $122s(n) + O(n)$ since there are 122 calls to C. Next, we put $10^4 n$ independent copies together in parallel and output the majority vote. Let X_i be the random variable that the outcome of the ith copy is $\Pi_{i=1}^{n} w_i$. By Lemma 14, on every input the new circuit computes FWP_n with probability at least $1 - \frac{120^{-n}}{2}$. Thus there is a random sequence that can be hard-wired in to the circuit, with the property that the resulting circuit gives the correct output on *every* input (and in fact, at least half of the random sequences have this property). This yields a deterministic TC^0 circuit computing FWP_n exactly which is of depth at most $d + c$ and of size no more than $(122 * 10^4) n(s(n) + cn)$ for some universal constant c . Choosing $\delta \geq (122 * 10^4)$ completes the proof. □

The problem FWP is strongly downward self-reducible [AK10, Definition , Proposition 7]. Hence, its worst-case hardness against TC^0 circuit families can be amplified as observed by Allender and Koucký [AK10, Corollary 17].

Theorem 18 *[AK10] If there is a* $\gamma > 0$ *such that* FWP $\notin$ io-$\text{TC}^0(n^{1+\gamma})$*, then* FWP $\notin$ io-TC^0.

(Theorem 18 is not stated in terms of io-TC^0 in [AK10], but the proof shows that if there are infinitely many input lengths n where FWP has circuits of of size n^k, then there are infinitely many input lengths m where FWP has circuits of size $m^{1+\gamma}$. The strong downward self-reducibility property allows small circuits for inputs of size m to be constructed by efficiently using circuits for size $n < m$ as subcomponents.)

Since FWP is equivalent to WP via linear-size reductions on the same input length, the following corollary is its easy consequence.

Corollary 19 *If there is a* $\gamma > 0$ *such that* WP $\notin$ io-TC$^0(n^{1+\gamma})$*, then* FWP $\notin$ io-TC^0.

Combining Corollary 19 with Theorem 17 yields the average-case hardness of BWP from nearly-linear-size worst-case lower bounds for WP against TC^0 circuit families.

Corollary 20 *There exists a constant* $\epsilon > 0$ *such that if* $\exists\gamma > 0$ *such that* WP $\notin$ io-TC$^0(n^{1+\gamma})$*, then for any* k *and* d *there exists* $n_0 > 0$ *such that when* $n \geq n_0$*,* BWP$_n$ *is* $(1-\epsilon)$*-hard for any* TC^0 *circuit of size* n^k *and depth* d*.*

Define the following Boolean function WPM$_n$: $S_5^n \times S_5^{60} \to \{0,1\}$, where WPM$_n$ stands for Word Problem over Multi-set.

Definition 21 *The input to* WPM$_n$ *is a pair* $\langle w_1 w_2..w_n, v_1 v_2..v_{60}\rangle$*, where* $\forall i \in [1..n]$, $w_i \in S_5$ *and* $\forall j \in [1..60]$, $v_i \in S_5$. $\langle w_1 w_2..w_n, v_1 v_2..v_{60}\rangle \in$ WPM *if and only if* $\exists j \in [1..60]$, $\Pi_{i=1}^n w_i = v_j$.

BWP is the restriction of WPM$_n$ to the case where all v_is are distinct. Hence, WPM inherits the average-case hardness of BWP, since any circuit that computes WPM$_n$ on a sufficiently large fraction of inputs also approximates BWP well. Formally,

Lemma 22 *There is an absolute constant* $0 < c < 1$ *such that for every* $\epsilon > 0$*, if* BWP$_n$ *is* $(1-\epsilon)$*-hard for* TC^0 *circuits of size* n^k *and depth* d*, then* WPM$_n$ *is* $(1-c\epsilon)$*-hard for* TC^0 *circuits of size* n^k *and depth* d*.*

Proof. Let $c = \frac{\binom{120}{60}}{(120)^{60}}$. Note that c is the probability that a sequence of 60 permutations contains no duplicates and is in sorted order. Suppose there is a circuit C with the property that $Pr_{x \in S^n \times S^{60}}[C(x) \neq \text{WPM}(x)] \leq c\epsilon$. Then the conditional probability that $C(x) \neq$ WPM(x) given that the last 60 items in x give a list in sorted order with no duplicates is at most ϵ. This yields a circuit having the same size, solving BWP with error at most ϵ, using the uniform distribution over its domain, contrary to our assumption. □

Corollary 23 *There exists a constant $\epsilon > 0$ such that if $\exists \gamma > 0$ such that* WP $\notin$ io-TC$^0(n^{1+\gamma})$*, then for any k and d there exists $n_0 > 0$ such that when $n \geq n_0$,* WPM$_n$ *is $(1-\epsilon)$-hard for* TC^0 *circuits of size n^k and depth d.*

Yao's XOR lemma [Yao82] is a powerful tool to boost average-case hardness. We utilize a specialized version of the XOR lemma for our purpose. Several proofs of this useful result have been published. For instance, see the text by Arora and Barak [AB09] for a proof that is based on Impagliazzo's hardcore lemma [Imp95]. For our application here, we need a version of the XOR lemma that is slightly different from the statement given by Arora and Barak. In the statement of the lemma as given by them, g is a function of the form $\{0,1\}^n \to \{0,1\}$. However, their proof works for any Boolean function g defined over any finite alphabet, because both the hardcore lemma and its application in the proof of the XOR lemma are insensitive to the encoding of the alphabet. Hence, we state the XOR Lemma in terms of functions over an alphabet set Σ.

For any Boolean function g over some domain Σ^n, define $g^{\oplus m} : \Sigma^{nm} \to \{0,1\}$ by $g^{\oplus m}(x_1, x_2, .., x_m) = g(x_1) \oplus g(x_2) \oplus .. \oplus g(x_m)$ where $\oplus$ represents the parity function.

Lemma 24 *[Yao82] Let $\frac{1}{2} < \epsilon < 1$, $k \in \mathbb{N}$ and $\theta > 2(1-\epsilon)^k$. There is a constant $c > 1$ that depends only on $|\Sigma|$ such that if g is $(1-\epsilon)$-hard for* TC^0 *circuits of size s and depth d, then $g^{\oplus k}$ is $(\frac{1}{2} + \theta)$-hard for* TC^0 *circuits of size $\frac{\theta^2 s}{cn}$ and depth $d-1$.*

Let $\Sigma = S_5$. The following corollary is an immediate consequence of Corollary 23 and Lemma 24.

Corollary 25 *If there is a $\gamma > 0$ such that* WP $\notin$ io-TC$^0(n^{1+\gamma})$*, then for any k, k' and d there exists $n_0 > 0$ such that when $n \geq n_0$* (WPM$_n$)$^{\oplus n}$ *is $(\frac{1}{2} + \frac{1}{n^{k'}})$-hard for* TC^0 *circuits of size n^k and depth d.*

Let WP$^{\otimes} = \cup_{n \geq 1}\{x \mid (\text{WPM}_n)^{\oplus n}(x) = 1\}$. Note that it is a language in uNC^1 and, moreover, it is decidable in linear time.

Theorem 26 *If there is a $\gamma > 0$ such that* WP $\notin$ io-TC$^0(n^{1+\gamma})$*, then for any integer $k > 0$,* WP$^{\otimes}$ *is $(\frac{1}{2} + \frac{1}{n^k})$-hard for* TC^0.

2.3.2 Worst-case to Average-case Reduction for L

Here we show a similar worst-case to average-case connection as in the previous subsection, but for the class L which contains NC^1. Just as the word problem WP is complete for NC^1, the word problem PWP for S_n is complete for L [MC87].

Definition 27 *The language* PWP *consists of all inputs* $< w_1, w_2 \ldots w_n >$*, where each* w_i *encodes a permutation over* S_n *and* $\Pi_{i=1}^{n} w_i$ *is the identity permutation.*

We will use a few different encodings of permutations. Encoding 1 is where the permutation is represented simply as an ordered list of n distinct numbers between 1 and n - the interpretation of this list as a permutation is that if the k'th element in the list is j, then k maps to j in the permutation. Encoding 2 is less economical and represents a permutation as an ordered list of n ordered pairs $(i, \sigma(i))$, where i ranges from 1 to n and σ is a permutation on $[n]$. The interpretation here is that the i maps to $\sigma(i)$ in the permutation σ. Here, whether the list is ordered does not matter - all permutations of the ordered list represent the same permutation. The fact that each *pair* is ordered is of course critical.

Using the fact that Sorting is in TC^0, we can convert from Encoding 1 to Encoding 2 or vice versa in TC^0. The conversion from Encoding 1 to Encoding 2 is trivial - simply prefix each number in the ordered list by its index in the list. To convert from Encoding 2 to Encoding 1, sort using the first element of the ordered pair as the key, and retain only the second element in the sorted list.

For technical reasons, we will use a third even more verbose encoding - Encoding 3. In Encoding 3, a permutation σ is represented as an ordered list of n integers each of which is n bits long. The permutation represented by this list is the identity permutation if there are two elements of the list which are equal, and is otherwise the permutation σ where $\sigma(i)$ is the rank of the i'th element in the list, i.e., its index in the sorted order. Note that a permutation in Encoding 1 can be trivially converted to Encoding 3 by prefixing each element by $n - \log(n)$ zeroes. To convert from Encoding 3 to Encoding 1 in TC^0, first check that there are no "collisions" in the list, i.e., a pair of identical elements. If there is a collision, output the identity permutation - this can be done in AC^0. If there are no collisions, transform the ordered list to an ordered list of ordered pairs formed by pairing each element of the original list with its index in the list.

Sort according to the elements of the original list, but retain only the corresponding order on the indices. If the list survives the collision check, this yields a permutation in Encoding 1.

Using the fact that the composition of two TC^0 functions is in TC^0, we get that we can convert from Encoding 2 to Encoding 3 and vice versa in TC^0.

By default, we will consider the third encoding to be in effect. If this is not the case, we will explicitly say so.

For the purpose of studying a worst-case to average-case connection for L,we need a balanced version of the language PWP.

Definition 28 *The language* BPWP *consists of all inputs* $< w_1, w_2 \dots w_n, i >$*, where each* w_i *encodes a permutation over* S_n *and the the* i*'th bit of the encoding of* $\Pi_{i=1}^{n} w_i$ *according to Encoding 1 is 1.*

We assume a natural Boolean encoding of the inputs, where the only relevant inputs are of size $n^3 + \log(n) + \log\log(n)$, with n blocks of n^2 bits each representing $w_1 \dots w_n$ according to Encoding 3 and the last block representing i. We assume wlog that n is a power of 2 - BPWP remains complete for L with this restriction.

Lemma 29 *There is a family* $\{C_n\}$ *of randomized* TC^0 *circuits of polynomial size such that for each* n*, the output of* C_n *is* $O(n^2/2^n)$*-close in statistical distance to the uniform distribution over* (σ, σ^{-1})*, where* σ *is uniformly chosen in* S_n*. Moreover, when considered purely as bit strings, the first and second outputs* C_n *are* $O(n^2/2^n)$*-close to the uniform distribution.*

Proof.

The circuits C_n are defined as follows. First n numbers $x_1, x_2 \dots x_n$, with each $x_i, 1 \leq i \leq n$ being n bits long are generated at random. As per Encoding 3, this n-tuple of numbers represents a permutation σ. The identity permutation is generated with probability at most $1/n! + n^2/2^n$, since the probability of a collision is at most $n^2/2^n$. Every other permutation is generated with equal probability, which is at least $(1 - n^2/2^n)1/n!$. A simple computation of the statistical distance yields that the corresponding distribution on permutations is $O(n^2/2^n)$-close to the uniform distribution on permutations σ over $[n]$.

It now remains to show how to generate σ^{-1}. Sort the x-list $x_1, x_2 \ldots x_n$ - this can be done in TC^0. Then convert σ from Encoding 3 to Encoding 2 in TC^0. Then we include circuitry which *reverses* the order of each ordered pair in the list, to yield the representation of σ^{-1} according to Encoding 2. Then implement the TC^0 conversion from Encoding 2 to Encoding 1, and finally use the elements of the resulting list as ranks to select elements from the sorted x-list. We thus derive a representation of σ^{-1} according to Encoding 3 which is itself a permutation of the representation of σ according to Encoding 3. The last part of the lemma follows using this fact and the argument in the previous paragraph on the relative unlikelihood of the identity permutation being represented. □

Lemma 29 gives us the ability to generate a random permutation and its inverse efficiently. This can be used to implement a random self-reduction in TC^0 and hence derive a worst-case to average-case hardness amplification in L against TC^0.

Theorem 30 *If* $L \not\subseteq TC^0$*, then there is a language in* L *which is* $(1 - 1/n^2)$*-hard for* TC^0.

Proof. The language for which we show a random self-reduction is BPWP. Assume that BPWP is $(1 - 1/n^2)$-easy for TC^0. We show how to solve BPWP in TC^0 based on this assumption. Since BPWP is complete for L, this implies that $L \subseteq TC^0$.

Let $< w_1, w_2 \ldots w_n, i >$ be an input to BPWP, where each w_i represents a permutation over S_n according to Encoding 3. We generate $n \log(n)$ randomized queries to BPWP such that for each query, the query with the last co-ordinate omitted is $1 - O(n^2/2^n)$-close to the uniform distribution over binary strings. The queries are generated in TC^0 as follows. Using Lemma 29, generate n random permutations $\sigma_1, \sigma_2 \ldots \sigma_n$ and their inverses. We do not know how to do this exactly, but it is enough to do it approximately as guaranteed by Lemma 29. Form the permutations $s_1, s_2 \ldots s_n$, where for each $i, 1 < i < n$, $s_i = \sigma_{i-1}^{-1} w_i \sigma_i$, $s_1 = w_1 \sigma_1$ and $s_n = w_n \sigma_n$. To form these permutations, convert to Encoding 1 and use the fact that two permutations can be multiplied in TC^0 when represented in Encoding 1. When converting back to Encoding 3, for each $i, 1 \leq i \leq n$, sort the list of numbers representing σ_i and then use the representation of the permutation in Encoding 1 as ranks to select from the sorted list. Thus for each i, the resulting permutation is exponentially close to a random permutation of the list of numbers representing σ_i. Since the σ_i are all independent, we have that $s_1 \ldots s_n$ are all

independent and exponentially close to the uniform distribution as bit strings. Now form the queries $< s_1, s_2 \ldots s_n, j >$ for each $1 \leq j \leq n \log(n)$.

Since BPWP is $(1 - 1/n^2)$-easy for TC^0 and by the assumption on the distribution of queries, we have that the TC^0 approximators for BPWP return the correct answers for all queries with probability at least $1 - 1/n$, for large enough n. Using the correct answers for all queries, we can reconstruct $s_1 s_2 \ldots s_n$ in Encoding 1. Also we know that $w_1 w_2 \ldots w_n = s_1 s_2 \ldots s_n \sigma_n^{-1}$. Thus we can reconstruct $w_1 w_2 \ldots w_n$ in Encoding 1 with another multiplication in TC^0 and thereby obtain the i'th bit with high probability. By a standard amplification step and using Adleman's trick [Adl78], this probabilistic circuit can be converted to a non-uniform one. □

2.3.3 Worst-case to Average-case Reduction for GapL and GapNC^1

We first consider GapL. Let $\mathrm{Determinant}$ denote the problem of computing the integer determinant. This is a complete problem for GapL (see, e.g. [MV97]). We show that if $\mathrm{Determinant}$ cannot be computed by TC^0 circuits then $\mathrm{Determinant}$ is somewhat hard on average for TC^0 circuits. As TC^0 circuits take Boolean input, we will encode each integer entry of an $n \times n$ integer matrix in binary. In order to keep the overall size of this Boolean input bounded, we will make the simplifying assumption that each entry of an $n \times n$ integer matrix instance of $\mathrm{Determinant}$ is at most n bits long. It is not hard to see that this version of $\mathrm{Determinant}$ too is complete for GapL. Since the proof of the next theorem is similar to the standard argument for proving random self-reducibility of $\mathrm{Permanent}$ [Lip91], we omit some low-level details.

Theorem 31 *Let $\mathcal{M}_n$ denote all $n \times n$ integer matrices with integer entries of size at most n bits. If there is a* TC^0 *circuit computing* $\mathrm{Determinant}$ *for at least $1 - \frac{1}{n^5}$ fraction of inputs from $\mathcal{M}_n$ then there is a* TC^0 *circuit that computes* $\mathrm{Determinant}$ *for all inputs from $\mathcal{M}_n$.*

Proof. Let C' denote the TC^0 circuit that computes the integer determinant for $1 - \frac{1}{n^5}$ fraction of inputs from $\mathcal{M}_n$. Our goal is to construct a TC^0 circuit that computes the integer determinant for *every* input matrix $M \in \mathcal{M}_n$. For input $M \in \mathcal{M}_n$, we will describe a *nonadaptive* reduction from the problem of computing $\det(M)$ to computing $\det(M_i)$ for a sequence of random matrices $M_i \in \mathcal{M}_n, 1 \leq i \leq r$ where each M_i is nearly uniformly distributed in $\mathcal{M}_n$.

To this end, pick a random matrix $A \in \mathcal{M}_n$. This requires $n^3 + n^2$ independent unbiased coin flips to pick the n^2 random n-bit entries of A along with their signs. Now, consider the polynomial $\det(M + Ax)$. This is a degree n polynomial over $\mathbb{Z}$ in the indeterminate x. Let $S = \{1, 2, \ldots, n+1\}$ be distinct interpolating points and consider $\det(M+Ai)$ for each $i \in S$. The matrix $M + Ai$ is random. Unfortunately, it is not uniformly distributed in $\mathcal{M}_n$ (indeed, even its support is not contained in $\mathcal{M}_n$). Therefore, we cannot directly use the circuit C' to compute $\det(M + Ai)$ for all $i \in S$ and interpolate the value of $\det(M)$. We shall get around this difficulty with Chinese remaindering.

By the Hadamard bound $|\det(M)| \leq 2^{n^2} \cdot n! < 4^{n^2}$ for all $M \in \mathcal{M}_n$. We can pick n^2 distinct $O(\log n)$ bit primes $p_1, p_2, \ldots, p_{n^2}$ so that $\prod_i p_i > |\det(M)|$ for each $M \in \mathcal{M}_n$. We note that $\det(M)$ can be reconstructed from the residues $\det(M) (\mathrm{mod}\, p_i), 1 \leq i \leq n^2$ by Chinese remaindering and, moreover, this reconstruction can be done in Dlogtime-uniform TC^0 [HAB02]. Hence, it suffices to describe a TC^0 circuit family for computing $\det(M) (\mathrm{mod}\, p)$ for each $M \in \mathcal{M}_n$, where p is an $O(\log n)$ bit prime.

For a matrix $A \in \mathcal{M}_n$ picked uniformly at random, consider $\det(M + Ax) (\mathrm{mod}\, p)$. This is a degree n polynomial in x modulo p. We will compute $\det(M) (\mathrm{mod}\, p)$ by interpolation. Let $S = \{1, 2, \ldots, n + 1\}$ be the distinct interpolating points in $\mathbb{F}_p$ (to ensure that this yields more than n points for all the primes p_i we will pick $p_i > n$ for all i). For any fixed $s \in S$ the matrix $M + As (\mathrm{mod}\, p)$ is nearly uniformly distributed over $n \times n$ matrices with $\mathbb{F}_p$ entries (the error, i.e. the statistical distance to uniform, can be bounded by $2^{-O(n)}$). However, as explained above, notice that we cannot directly use the circuit C' to compute $\det(M + As)$ since the entries of $M + As$ can be $O(n + \log n)$ bits long. Neither can we directly use C' to compute $\det(M + As) (\mathrm{mod}\, p)$, because the matrix $M + As (\mathrm{mod}\, p)$ has integer entries in the range $\{0, 1, \ldots, p - 1\}$ and these matrices are only a $(\frac{p}{2^n})^{n^2}$ fraction of matrices in $\mathcal{M}_n$. It is possible that the output of C' is incorrect on all these matrices. We now describe the solution. Consider the mapping

$$f : \mathcal{M}_n \longrightarrow \mathbb{F}_p^{n \times n},$$

where $f(M) = M (\mathrm{mod}\, p)$. This is an onto mapping such that first picking a uniformly distributed random matrix $M' \in \mathbb{F}_p^{n \times n}$ and then picking a uniformly distributed random matrix

$M \in f^{-1}(M')$ results in M being nearly uniform in $\mathcal{M}_n$ (the error is bounded by $2^{-O(n)}$). Furthermore, given M' it is easy to sample nearly uniformly from $f^{-1}(M')$ by adding a suitable random multiple of p to each entry of M'.

Now, for the random matrix $M + As(\mathrm{mod}\, p) \in \mathbb{F}_P^{n\times n}$ let M_s denote this random preimage in $\mathcal{M}_n$, for $s \in \{1, 2, \ldots, n+1\}$. By the above argument it follows that each M_s is nearly uniformly distributed in $\mathcal{M}_n$. Hence, for each $s \in S$:

$$\Pr[C'(M_s) = \det(M_s)] \geq 1 - \frac{1}{n^4}.$$

Hence with probability $1 - \frac{1}{n^3}$ the circuit C' correctly computes $\det(M_s)$ for all $s \in S$. Now, applying the fact that polynomial interpolation is TC^0 computable [HAB02], a TC^0 circuit can recover $\det(M)(\mathrm{mod}\, p)$, given $\det(M_s)(\mathrm{mod}\, p)$ for all $s \in S$.

Putting it together, for each prime p_i we have a randomized TC^0 circuit that computes $\det(M)(\mathrm{mod}\, p_i)$ with probability $1 - \frac{1}{n^3}$. Finally, applying Chinese remaindering which is TC^0 computable [HAB02], we obtain a randomized TC^0 circuit that computes $\det(M)$ with probability $1 - \frac{1}{n}$. The random bits can be fixed after amplifying the success probability using standard techniques. □

We now briefly discuss a similar worst-case to average-case reduction for GapNC^1. The problem of computing the $(1,1)^{th}$ entry of the product of n 3×3 integer matrices is GapNC^1 complete [CMTV98]. We show that if this problem cannot be computed by TC^0 circuits then it is somewhat hard on average for TC^0 circuits. As before, since we consider TC^0 circuits which take Boolean input, we consider input $(M_1, M_2, \cdots, M_n)$ in a smaller set $\mathcal{I}_n$ such that each M_i is a 3×3 matrix with integer entries that are at most n bits long. This restricted problem is also easily seen to be GapNC^1 complete. In order to show the worst-case to average-case reduction we pick a uniform random instance $(A_1, A_2, \cdots, A_n) \in \mathcal{I}_n$ and consider the instance $(M_1 + A_1x, M_2 + A_2x, \cdots, M_n + A_nx)$ for indeterminate x. Notice that the $(1,1)^{th}$ entry of the matrix $\prod_{i=1}^{n}(M_i + A_ix)$ is a degree n polynomial in x. Now, exactly along the same lines as the proof of Theorem 31 we can show the following.

Theorem 32 *Let $\mathcal{I}_n$ denote all iterated matrix multiplication instances $M_1, M_2, \cdots, M_n$, consisting of 3×3 integer matrices M_i whose entries are at most n bits long. If there is a* TC^0

circuit computing $\prod_{i=1}^{n} M_i$ *for at least* $1 - \frac{1}{n^5}$ *inputs* $M_1, M_2, \cdots, M_n$ *in* $\mathcal{I}_n$ *then there is a* TC^0 *circuit that computes* $\prod_{i=1}^{n} M_i$ *for all inputs* $M_1, M_2, \cdots, M_n$ *in* $\mathcal{I}_n$.

2.4 Uniform derandomization

The Nisan-Wigderson generator is the canonical method to prove the existence of pseudo-random generators based on hard functions. It relies on the following definition of combinatorial designs.

Definition 33 (Combinatorial Designs) *Fix a universe of size* u. *An* (m, l)*-design of size* n *on* $[u]$ *is a list of subsets* $S_1, S_2, ..., S_n$ *satisfying:*

1. $\forall i \in [1..n],\ |S_i| = m$;
2. $\forall i \neq j \in [1..n],\ |S_i \cap S_j| \leq l$.

Nisan and Wigderson [NW94] invented a general approach to construct combinatorial designs for various ranges of parameters. The proof given by Nisan and Wigderson gives designs where $l = \log n$, and most applications have used that value of l. For our application, l can be considerably smaller, and furthermore, we need the S_i's to be very efficiently computable. For completeness, we present the details here. (Other variants of the Nisan-Wigderson construction have been developed for different settings; we refer the reader to one such construction by Viola [Vio05], as well as to a survey of related work [Vio05, Remark 5.3].)

Lemma 34 *[vL99] For* $l > 0$, *the polynomial* $x^{2 \cdot 3^l} + x^{3^l} + 1$ *is irreducible over* $\mathbb{F}_2[x]$.

Lemma 35 *[NW94] For any integer* n, *any* α *such that* $\log \log n / \log n < \alpha < 1$, *let* $b = \lceil \alpha^{-1} \rceil$ *and* $m = \lceil n^{\alpha} \rceil$, *there is a* (m, b)*-design with* $u = O(m^6)$. *Furthermore, each* S_i *can be computed within* $O(bm^2)$ *time.*

Proof. Fix $q = 2^{2 \cdot 3^l}$ for some l such that $m \leq q \leq m^3$. Let the universe be $\mathbb{F}_q \times \mathbb{F}_q$ and S_i be the graph of the ith univariate polynomial of degree at most b in the standard order. Since $q^b \geq (n^{\alpha})^b \geq n$, there are at least n distinct S_is. No two polynomials share more than b points, hence, the second condition of Definition 33 is satisfied. The first condition holds because we could simply drop elements without increasing the size of intersections.

The arithmetic operations in $\mathbb{F}_q$ are performed within $\log^{O(1)} q$ time because of the explicitness of the irreducible polynomial by Lemma 34. It is evident that for any $i \in [n]$, we are able to enumerate all elements of S_i in time $O(m \cdot b(\log^{O(1)} q)) = O(bm^2)$. □

Lemma 36 *For any constant $\alpha > 0$ and for any large enough integer n, if g is $(\frac{1}{2} + \frac{1}{n^2})$-hard for* TC^0 *circuits of size n^2 and depth $d + 2$, then any probabilistic* TC^0 *circuit C of size n and depth d can be simulated by another probabilistic* TC^0 *circuit of size $O(n^{1+\alpha})$ and depth $d+1$ which is given oracle access to $g_{\lceil n^\alpha \rceil}$ and uses at most $O(n^{6\alpha})$ many random bits.*

Proof. This is a direct consequence of Lemma 35; we adapt the traditional Nisan-Wigderson argument to the setting of TC^0 circuits. Let n and α be given, with $0 < \alpha < 1$. Let $S_1, \ldots, S_n$ be the (m, b)-design from Lemma 35, where $m = \lceil n^\alpha \rceil$, $b = \lceil \alpha^{-1} \rceil$, and each $S_i \subset [u]$, with $u = O(m^6)$. We are given $g : \Sigma^m \to \{0,1\}$; define $h^g : \Sigma^u \to \{0,1\}^n$ by $h^g(x) = g(x|_{S_1})g(x|_{S_2})..g(x|_{S_n})$, where $x|_{S_i}$ is the sub-sequence restricted to the coordinates specified by S_i.

The new circuit samples randomness uniformly from A^u and feeds C with pseudo-random bits generated by h^g instead of purely random bits. It only has one more extra layer of oracle gates and its size is bounded by $O(n + n * n^\alpha) = O(n^{1+\alpha})$. What is left is to prove the following claim.

Claim 37 *For any constant $\epsilon > 0$, $|Pr_{x \in_U A^u}[C(h^g(x)) = 1] - Pr_{y \in_U \{0,1\}^n}[C(y) = 1]| < \epsilon$.*

Proof. Suppose there exists ϵ such that $|Pr_{x \in \{0,1\}^n}[C(x) = 1] - Pr_{y \in A^n}[C(h^g(y)) = 1]| \geq \epsilon$. We will seek a contradiction to the hardness of g via a hybrid argument.

Sample z uniformly from A^n and r uniformly from $\{0,1\}^n$. Create a sequence of $n+1$ distributions H_i on $\{0,1\}^n$ where

- $H_0 = r$;
- $H_n = h^g(z)$;
- $\forall 1 \leq i \leq n-1,\ H_i = h^g(z)_1 h^g(z)_2 \ldots h^g(z)_i r_{i+1} \ldots r_n$.

By our assumption, $|\Sigma_{j=1}^{n}(Pr_{x\sim H_{j-1}}[C(x)=1]-Pr_{x\sim H_j}[C(x)=1])|\geq\epsilon$. Therefore, $\exists j\in[n]$ such that $|Pr_{x\sim H_{j-1}}[C(x)=1]-Pr_{x\sim H_j}[C(x)=1]|\geq\frac{\epsilon}{n}$. Let i be one such index.

Assume $Pr_{x\sim H_i}[C(x)=1]-Pr_{x\sim H_{i-1}}[C(x)=1]\geq\frac{\epsilon}{n}$, otherwise add a not gate at the top of C, and treat the new circuit as C instead.

Consider the following probabilistic TC^0 circuit C' for g. On input x, sample z uniformly from A^n and r uniformly from $\{0,1\}^n$, replace the coordinates of z specified by S_i with x. Sample a random bit $b\in\{0,1\}$. If $C(h^g(z)_1\dots h^g(z)_{i-1}br_{i+1}\dots r_n)=1$, output b, otherwise, output $1-b$.

$$\begin{aligned}
& Pr_{x\in A^{n^\alpha}}[C'(x)=f(x)]\\
= \;& \tfrac{1}{2}Pr_{x\in A^{n^\alpha}}[C'(x)=b\mid b=f(x)]+\tfrac{1}{2}Pr_{x\in A^{n^\alpha}}[C'(x)\neq b\mid b\neq f(x)]\\
= \;& \tfrac{1}{2}Pr_{x\in A^{n^\alpha}}[C'(x)=b\mid b=f(x)]+\tfrac{1}{2}-\tfrac{1}{2}Pr_{x\in A^{n^\alpha}}[C'(x)=b\mid b\neq f(x)]\\
= \;& \tfrac{1}{2}+\tfrac{1}{2}Pr_{x\in A^{n^\alpha}}[C'(x)=b\mid b=f(x)]-\tfrac{1}{2}Pr_{x\in A^{n^\alpha}}[C'(x)=b\mid b\neq f(x)]\\
= \;& \tfrac{1}{2}+Pr_{x\in A^{n^\alpha}}[C'(x)=b\mid b=f(x)]-Pr_{x\in A^{n^\alpha}}[C'(x)=b]\\
= \;& \tfrac{1}{2}+(Pr_{y\in H_i}(C(y)=1)-Pr_{y\in H_{i-1}}(C(y)=1))\\
\geq \;& \tfrac{1}{2}+\tfrac{\epsilon}{n}
\end{aligned}$$

Hence, there is a fixing of values for z, r and b satisfying the property that $Pr_{x\in A^{n^\alpha}}[C'(x,z,r,b)=f(x)]\geq\frac{1}{2}+\frac{\epsilon}{n}$. Note that in this case $\forall 1\leq k\leq i-1,\ h^g(z)_k$ is function on input $x|_{S_k\cap S_i}$. Since $\forall k\neq i,\ |S_i\cap S_k|\leq b$, we only need a TC^0 circuit of size at most $2^{O(b)}$ and of depth at most 2 to compute each $h^g(z)_k$. In conclusion, we obtain a TC^0 circuit C'' of size at most $(2^{O(b)}+1)n$ and of depth at most $d+2$ such that $Pr_{x\in A^{n^\alpha}}[C'(x)=f(x)]\geq\frac{1}{2}+\frac{\epsilon}{n}\geq\frac{1}{2}+\frac{1}{n^2}$ when n is large enough, a contradiction. □

□

The simulation in Lemma 36 is quite uniform, thus, plugging in appropriate segments of $\mathrm{WP}^{\otimes}$ as our candidates for the hard function g, we derive our first main result.

Theorem 38 *[Theorem 7 restated.] If* WP *is not infinitely often computed by* $\mathrm{TC}^0(n^{1+\gamma})$ *circuit families for some constant* $\gamma>0$*, then any language accepted by polynomial-size probabilistic uniform* TC^0 *circuit family is in* $\mathsf{uTC}^0(\mathsf{SUBEXP})$.

Proof. Fix any small constant $\delta>0$. Let L be a language accepted by some probabilistic uniform TC^0 circuit family of size at most n^k and of depth at most d for some constants k,d.

Choose m such that $n^{\frac{\delta}{12}} \leq m \leq n^{\frac{\delta}{6}}$, and let α be such that $m = n^{\alpha}$. By Theorem 26, when m is large enough, $\mathrm{WP}_m^{\otimes}$ is $(\frac{1}{2} + \frac{1}{n^{2k}})$-hard for TC^0 circuits of size n^{2k} and depth $d + c$, where c is any constant. Hence, as a consequence of Lemma 36, we obtain a probabilistic oracle TC^0 circuit for L_n of depth $d + 1$. Since the computation only needs $O(m^6)$ random bits, it can be turned into a deterministic oracle TC^0 circuit of depth $d + 2$ and of size at most $O(n^{2k}) * 2^{O(m^6)} \leq 2^{O(n^{\delta})}$ (when n is large enough), where we evaluate the previous circuit on every possible random string and add an extra MAJORITY gate at the top. The oracle gates all have fan-in $m \leq n^{\delta/6}$, and thus can be replaced by DNF circuits of size $2^{O(n^{\delta})}$, yielding a deterministic TC^0 circuit of size $2^{O(n^{\delta})}$ and depth $d + 3$.

We need to show that this construction is uniform, so that the direct connection language can be recognized in time $O(n^{\delta})$. The analysis consists of three parts.

- The connectivity between the top gate and the output gate of individual copies is obviously computable in time $m^6 \leq n^{\delta}$.
- The connectivity inside individual copies is $\mathsf{DLOGTIME}$-uniform, hence, n^{δ}-uniform.
- By Lemma 35 each S_i is computable in time $O(dm^2)$ which is $O(m^2)$ since d is a constant only depending on δ. Moreover, notice that $\mathrm{WP}^{\otimes}$ is a linear-time decidable language. Therefore, the DNF expression corresponding to each oracle gate can be computed within time $O(m^2) \leq n^{\delta}$.

In conclusion, the above construction produces a uniform TC^0 circuit of size $2^{n^{\delta}}$. Since δ is arbitrarily chosen, our statement holds. □

Note that the above conclusion can be strengthened to the following form: any language accepted by a polynomial-size probabilistic $o(n)$-uniform TC^0 circuit family is in $\mathsf{uTC}^0(\mathsf{SUBEXP})$.

2.5 Consequences of pathetic arithmetic circuit lower bounds

In this section we show that a pathetic lower bound assumption for *arithmetic circuits* yields a uniform derandomization of a special case of polynomial identity testing (introduced and studied by Dvir *et al.* [DSY09]).

The explicit collection of polynomials that we consider is $\{\text{IMM}_{3,n}\}_{n>0}$, where $\text{IMM}_{3,n}$ is the $(1,1)^{th}$ entry of the product of n 3×3 matrices whose entries are all distinct indeterminates. Notice that $\text{IMM}_{3,n}$ is a degree n multilinear polynomial in $9n$ indeterminates, and $\text{IMM}_{3,n}$ can be considered as a polynomial over any field $\mathbb{F}$.

Arithmetic circuits computing a polynomial in the ring $\mathbb{F}[x_1, x_2, \ldots, x_n]$ are directed acyclic graphs with the indegree zero nodes (the inputs nodes) labeled by either a variable x_i or a scalar constant. Each internal node is either a $+$ gate or a $\times$ gate, and the circuit *computes* the polynomial that is naturally computed at the output gate. The circuit is a *formula* if the fanout of each gate is 1.

Before going further, we pause to clarify a point of possible confusion. There is another way that an arithmetic circuit C can be said to compute a given polynomial $f(x_1, x_2, \ldots, x_n)$ over a field $\mathbb{F}$; even if C does not compute f in the sense described in the preceding paragraph, it can still be the case that for all scalars $a_i \in \mathbb{F}$ we have $f(a_1, \ldots, a_n) = C(a_1, \ldots, a_n)$. In this case, we say that C *functionally* computes f over $\mathbb{F}$. If the field size is larger than the syntactic degree of circuit C and the degree of f, then the two notions coincide. Assuming that f is not *functionally* computed by a class of circuits is a *stronger* assumption than assuming that f is not computed by a class of circuits (in the usual sense). In our work in this paper, we use the weaker intractability assumption.

An *oracle* arithmetic circuit is one that has *oracle* gates: For a given sequence of polynomials $A = \{A_n\}$ as oracle, an oracle gate of fan-in n in the circuit evaluates the n-variate polynomial A_n on the values carried by its n input wires. An oracle arithmetic circuit is called *pure* (following [AK10]) if all non-oracle gates are of bounded fan-in. (Note that this use of the term "pure" is unrelated to the "pure" arithmetic circuits defined by Nisan and Wigderson [NW97].)

The class of polynomials computed by polynomial-size arithmetic formulas is known as arithmetic NC^1. By [BOC92] the polynomial $\text{IMM}_{3,n}$ is complete for this class. Whether $\text{IMM}_{3,n}$ has polynomial size *constant-depth* arithmetic circuits is a long-standing open problem in the area of arithmetic circuits [NW97]. In this context, the known lower bound result is that $\text{IMM}_{3,n}$ requires exponential size multilinear depth-3 circuits [NW97].

Very little is known about lower bounds for general constant-depth arithmetic circuits, compared to what is known about constant-depth Boolean circuits. Exponential lower bounds for depth-3 arithmetic circuits over finite fields were shown in [GK98] and [GR00]. On the other hand, for depth-3 arithmetic circuits over fields of characteristic zero only quadratic lower bounds are known [SW01]. However, it is shown in [RY09] that the determinant and the permanent require exponential size *multilinear* constant-depth arithmetic circuits. More details on the current status of arithmetic circuit lower bounds can be found in Raz's paper [Raz10, Section 1.3].

Definition 39 *We say that the polynomials $\{p_n\}_{n>0} \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ is $(s(n), m(n), d)$-downward self-reducible if there is a pure oracle arithmetic circuit C_n of depth $O(d)$ and size $O(s(n))$ that computes the polynomial p_n using oracle gates only for $p_{m'}$, for $m' \leq m(n)$.*

Analogous to [AK10, Proposition 7], we can easily observe the following. It is a direct divide and conquer argument using the iterated product structure.

Lemma 40 *For each $1 > \epsilon > 0$ the polynomial sequence $\{IMM_{3,n}\}$ is $(n^{1-\epsilon}, n^{\epsilon}, 1/\epsilon)$-downward self-reducible.*

An easy argument, analogous to Theorem 18, shows that Lemma 40 allows for the amplification of weak lower bounds for $\{\mathrm{IMM}_{3,n}\}$ against arithmetic circuits of constant depth:

Theorem 41 *Suppose there is a constant $\delta > 0$ such that for all d and every n, the polynomial sequence $\{IMM_{3,n}\}$ requires depth-d arithmetic circuits of size at least $n^{1+\delta}$. Then, for any constant depth d the sequence $\{IMM_{3,n}\}$ is not computable by depth-d arithmetic circuits of size n^k for any constant $k > 0$.*

Our goal is to apply Theorem 41 to derandomize a special case of polynomial identity testing (first studied in [DSY09]). To this end we restate a result of Dvir et. al [DSY09].

Theorem 42 (Theorem 4 in [DSY09]) *Let n, s, r, m, t, d be integers such that $s \geq n$. Let $\mathbb{F}$ be a field which has at least $2mt$ elements. Let $P(x, y) \in \mathbb{F}[x_1, \ldots, x_n, y]$ be a non-zero polynomial with $\deg(P) \leq t$ and $\deg_y(P) \leq r$ such that P has an arithmetic circuit of size*

s and depth d over $\mathbb{F}$. Let $f(x) \in \mathbb{F}[x_1, \ldots, x_n]$ be a polynomial with $\deg(f) = m$ such that $P(x, f(x)) \equiv 0$. Then $f(x)$ can be computed by a circuit of size $s' = poly(s, m^r)$ and depth $d' = d + O(1)$ over $\mathbb{F}$.

Let the underlying field $\mathbb{F}$ be large enough ($\mathbb{Q}$, for instance). The following lemma is a variant of Lemma 4.1 in [DSY09]. For completeness, we provide its proof here.

Lemma 43 (Variant of Lemma 4.1 in [DSY09]) *Let n, r, s be integers and let the polynomial $f \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ be a nonzero polynomial with individual degrees at most r that is computed by an arithmetic circuit of size $s \geq n$ and depth d. Let $m = n^\alpha$ be an integer where $\alpha > 0$ is an arbitrary constant. Let $S_1, S_2, \ldots, S_n$ be the sets of the (m, b)-design constructed in Lemma 35 where $b = \lceil \frac{1}{\alpha} \rceil$. Let $p \in \mathbb{F}[z_1, \ldots, z_m]$ be a multilinear polynomial with the property that*

$$F(y) = F(y_1, y_2, \ldots, y_u) \triangleq f(p(y|_{S_1}), \ldots, p(y|_{S_n})) \equiv 0 \tag{2.1}$$

Then there exists absolute constants a and k such that $p(z)$ is computable by an arithmetic circuit over $\mathbb{F}$ with size bounded by $O((sm^r)^a)$ and having depth $d + k$.

Proof. Consider the following set of hybrid polynomials:

$$\begin{aligned} F_0(x, y) &= f(x_1, x_2, \ldots, x_n) \\ F_1(x, y) &= f(p(y|_{S_1}), x_2, \ldots, x_n) \\ &\vdots \\ F_n(x, y) &= f(p(y|_{S_1}), \ldots, p(y|_{S_n})) \end{aligned}$$

The assumption implies that $F_0 \not\equiv 0$ while $F_n \equiv 0$. Hence, there exists $0 \leq i < n$ such that $F_i \not\equiv 0$ and $F_{i+1} \equiv 0$. Notice that F_i is a nonzero polynomial in the variables $\{x_j \mid i + 2 \leq j \leq n\}$ and the variables $\{y_j \mid j \in S_1 \cup S_2 \cup \cdots \cup S_i\}$.

We recall the well-known Schwartz-Zippel lemma.

Lemma 44 (Schwartz-Zippel) *Let $\mathbb{F}$ be a field and let $f \in \mathbb{F}[x_1, ..., x_n]$ be a non-zero polynomial with total degree at most r. Then for any finite subset $S \subset \mathbb{F}$ we have*

$$|\{c \in S^n \ : \ f(c) = 0\}| \leq r \cdot |S|^{n-1} \tag{2.2}$$

Since $\deg(F_i) \leq nrm$, then if we assume that $\mathbb{F}$ has size more than nrm, Lemma 44 assures that we can assign values from the field $\mathbb{F}$ to the variables $\{x_j \mid i+2 \leq j \leq n\}$ and the variables $\{y_j \mid j \notin S_{i+1}\}$ so that F_i remains a nonzero polynomial in the remaining variables. More precisely, fixing these variables to scalar values yields a polynomial $\tilde{f}$ with the property that

$$\begin{aligned} \tilde{f}(q_1(y|_{S_1 \cap S_{i+1}}), \ldots, q_1(y|_{S_i \cap S_{i+1}}), x_{i+1}) &\not\equiv 0 \\ \tilde{f}(q_1(y|_{S_1 \cap S_{i+1}}), \ldots, q_1(y|_{S_i \cap S_{i+1}}), p(y|_{S_{i+1}})) &\equiv 0 \end{aligned}$$

where $q_j(y|_{S_j \cap S_{i+1}})$ is the polynomial obtained from $p_j(y|_{S_j})$ after fixing the variables in $S_j \setminus S_{i+1}$.

Rename the variables $\{y_j \mid j \in S_{i+1}\}$ with $\{z_j \mid 1 \leq j \leq m\}$ and replace x_{i+1} by w. We obtain a polynomial g with the property that

$$\begin{aligned} g(z_1, \ldots, z_m, w) &\not\equiv 0 \\ g(z_1, \ldots, z_m, p(z_1, \ldots, z_m)) &\equiv 0 \end{aligned}$$

In order to apply Theorem 42, the only thing that remains is to calculate the circuit complexity of g. $\forall j \neq i+1,\ |S_j \cap S_{i+1}| \leq b$ which is a constant. Hence, for any $j \leq i$, $q_j(y|_{S_j \cap S_{i+1}})$ is a polynomial depending on a constant number of variables, which can be computed by a constant-size arithmetic circuit of depth 2 (Basically, it is a sum of monomials). Under the assumption that f has a circuit of size s and depth d, g is computable by a circuit of size $s + O(n)$ and depth $d+2$ which is a composition of the aforementioned circuits. It is important to note that $\deg_w(g) = \deg_{x_{i+1}}(f) \leq r$.

Now we use Theorem 42 to obtain that $p(z)$ has a circuit of size at most $(sm^r)^a$ and depth $d+k$, which concludes our proof. □

At this point we describe our deterministic black-box identity testing algorithm for constant-depth arithmetic circuits of polynomial size and bounded individual degree. Let n, m, u, α be the parameters as in Lemma 35. Given such a circuit C over variables $\{x_i \mid i \in [n]\}$ of size n^t, depth d and individual degree r, we simply replace x_i with $\text{IMM}_{3,m}(y|S_i)$ where y is a new set of variables $\{y_j \mid j \in [u]\}$. Let $\tilde{C}[y_1, \ldots, y_u]$ denote the polynomial computed by the new circuit.

Notice that the total degree of $\tilde{C}$ is bounded by u^c where c is a constant depending on the combinatorial design and r. Let $R \subseteq \mathbb{F}$ be any set of $u^c + 1$ distinct points. Then by Lemma 44 the polynomial computed by $\tilde{C}$ is identically zero if and only if $\tilde{C}(a_1, a_2, \ldots, a_u) = 0$ for all $(a_1, a_2, \ldots, a_u) \in R^u$.

This gives us the claimed algorithm. Its running time is bounded by $O((u^c + 1)^u) = O(2^{n^{7\alpha}})$. Since α can be chosen to be arbitrarily small, we have shown that this identity testing problem is in deterministic sub-exponential time. The correctness of the algorithm follows from the next lemma.

Lemma 45 *If for every constant $d' > 0$, the polynomial sequence $\{IMM_{3,n}\}$ is not computable by depth-d' arithmetic circuits of size n^k for any $k > 0$, then $C[x_1, \ldots, x_n] \equiv 0$ if and only if $\tilde{C}[y_1, \ldots, y_u] \equiv 0$.*

Proof. The only-if part is easy to see. Let us focus on the if part. Suppose it is not the case, which means that $\tilde{C}[y_1, \ldots, y_u] \equiv 0$ but $C[x_1, \ldots, x_n] \not\equiv 0$. Then let $C[x_1, \ldots, x_n]$ play the role of $f[x_1, \ldots, x_n]$ in Lemma 43 and let $\text{IMM}_{3,m}[z_1, \ldots, z_m]$ take the place of $p[z_1, \ldots, z_m]$. Therefore, $\text{IMM}_{3,m}[z_1, \ldots, z_m]$ is computable by a circuit of depth $d + k$ and size at most $(n^t m^r)^a = m^{O(1)}$, a contradiction. □

Putting it together, we get the following result.

Theorem 46 *If there exists $\delta > 0$ such that for any constant $e > 0$, $IMM_{3,n}$ requires depth-e arithmetic circuits of size at least $n^{1+\delta}$, then the black-box identity testing problem for constant-depth arithmetic circuits of polynomial size and bounded individual degree is in deterministic sub-exponential time.*

Next, we notice that the above upper bound can be sharpened considerably. The algorithm simply takes the OR over subexponentially-many evaluations of an arithmetic circuit; if any of the evaluations does not evaluate to zero, then we know that the expressions are not equivalent; otherwise they are. Note that evaluating an arithmetic circuit can be accomplished in logspace. (When evaluating a circuit over $\mathbb{Q}$, this is shown in [HAB02, Corollary 6.8]; the argument for other fields is similar, using standard results about the complexity of field arithmetic.) Note also that every language computable in logspace has AC^0 circuits of subexponential size. (This

appears to have been observed first by Gutfreund and Viola [GV04]; see also [AHM$^+$08] for a proof.) This yields the following uniform derandomization result.

Theorem 47 *[Theorem 9 restated.] If there are no constant-depth arithmetic circuits of size $n^{1+\epsilon}$ for the polynomial sequence $\{IMM_{3,n}\}$, then for every constant d, black-box identity testing for depth-d arithmetic circuits with bounded individual degree can be performed by a uniform family of constant-depth* AC^0 *circuits of subexponential size.*

We call attention to an interesting difference between Theorems 38 and 47. In Theorem 47, in order to solve the identity testing problem with uniform AC^0 circuits of size 2^{n^ϵ} for smaller and smaller ϵ, the depth of the AC^0 circuits increases as ϵ decreases. In contrast, in order to obtain a deterministic threshold circuit of size 2^{n^ϵ} to simulate a given probabilistic TC^0 algorithm, the argument that we present in the proof of Theorem 38 gives a circuit whose depth is not affected by the choice of ϵ. We do not know if a similar improvement of Theorem 47 is possible, but we observe here that the depth need not depend on ϵ if we use threshold circuits for the identity test.

Theorem 48 *If there are no constant-depth arithmetic circuits of size $n^{1+\epsilon}$ for the polynomial sequence $\{IMM_{3,n}\}$, then there is a constant c such that, for every constant d and every $\gamma > 0$, black-box identity testing for depth-d arithmetic circuits with bounded individual degree can be performed by a uniform family of depth $d + c$ threshold circuits of size 2^{n^γ}.*

Proof. We provide only a sketch. Choose $\alpha < \gamma/14$, where α is the constant from the discussion in the paragraph before Lemma 45. Thus, our identity testing algorithm will evaluate a depth d arithmetic circuit $C(x_1, \ldots, x_n)$ at fewer than $2^{n^{\gamma/2}}$ points $\vec{v} = (v_1, \ldots, v_n)$, where each v_i is obtained by computing an instance of $\mathrm{IMM}_{3,n^\alpha}$ consisting of n^α 3-by-3 matrices, whose entries without loss of generality have representations having length at most n^α. Thus these instances of $\mathrm{IMM}_{3,n^\alpha}$ have DNF representations of size $2^{O(n^{2\alpha})}$. These DNF representations are uniform, since the direct connection language can be evaluated by computing, for a given input assignment to $\mathrm{IMM}_{3,n^\alpha}$, the product of the matrices represented by that assignment, which takes time at most $(n^\alpha)^3 < \log(2^{n^{\gamma/2}})$. Evaluating the circuit C on $\vec{v}$ can be done in uniform TC^0 [AAD00, HAB02]. □

Chapter 3

Non-constant-depth Lower Bounds for NEXP against ACC circuits

In this chapter, we prove that NEXP does not have ACC_m circuits of quasi-polynomial size and $o(\log \log n)$ depth.

3.1 Preliminaries

General circuits consist of NOT gates and unbounded fan-in AND and OR gates. ACC_m circuits are general circuits equipped with unbounded fan-in MOD_m gates, where m is a fixed integer.

We say a Boolean function $g : \Sigma^n \to \{0,1\}$ is in $\text{ACC}_m(s,d)$ if g can be recognized by some ACC_m circuit of size at most s and depth bounded by d. For any two functions $s(n)$ and $d(n)$, we say a language $L \in \text{ACC}(s(n), d(n))$ if there exists an integer constant m such that for each input length n, its characteristic function L_n is in $\text{ACC}_m(s(n), d(n))$. For any two families of functions $\mathcal{S}$ and $\mathcal{D}$, $\text{ACC}(\mathcal{S}, \mathcal{D}) = \bigcup \text{ACC}(s(n), d(n)) \mid s(n) \in \mathcal{S}, d(n) \in \mathcal{D}$.

SYM^+ circuits have exactly two levels of internal nodes. The top level is a single gate with unbounded fan-in which computes an arbitrary symmetric function and the bottom level contains only AND gates which are connected directly to the input variables. We say a boolean function $g : \Sigma^n \to \{0,1\}$ is in $\text{SYM}^+(s,t)$ if it can be computed by some SYM^+ circuit of size at most s, where moreover, the fan-in of AND gates is bounded by t. We can define similarly as above the language classes $\text{SYM}^+(s(n), t(n))$ and $\text{SYM}^+(\mathcal{S}, \mathcal{T})$.

For a circuit type $\mathcal{C}$ and a set of associated measures, it will be convenient for us to consider the family of collections of boolean circuits which is denoted as $\text{Circuit}_{\mathcal{C}}(s_1(n), .., s_m(n)) = \{G_1, G_2, ..\}$, where each circuit in G_n has exactly n input variables and its ith measure is bounded by $s_i(n)$ respectively. For general circuits, we only consider the size measure, hence, $\text{Circuit}_{\text{General}}(s(n)) = \{G_1, G_2, ..\}$, where G_n contains all circuits of size at most $s(n)$. We can also give similar definitions for $\text{Circuit}_{\text{ACC}_m}(s(n), d(n))$ with both size and depth

measures and for $\text{Circuit}_{\text{SYM}^+}(s(n), t(n))$ where the first measure is the size measure and the second measure is in terms of the bottom fan-in.

For two families $\mathcal{F}_1 = \{G_1, G_2, ..\}$ and $\mathcal{F}_2 = \{G'_1, G'_2, ..\}$, we say $\mathcal{F}_1$ is *transformable to* $\mathcal{F}_2$ if for all sufficiently large $n \in \mathbb{N}$, $\forall C \in G_n$, $\exists C' \in G'_n$ such that $\forall x \in \Sigma^n, C(x) = C'(x)$, namely, C and C' are equivalent. Furthermore, $\mathcal{F}_1$ is *transformable to* $\mathcal{F}_2$ *in time* $t(n)$ if there exists a uniform algorithm which given the standard encoding of C, output C' in time $t(n)$.

For a family $\mathcal{F} = \{G_1, G_2, ..\}$, we say $\mathcal{F}$-SAT is *solvable in time* $t(n)$ if there exists a uniform algorithm $\mathcal{A}$ such that for all sufficiently large $n \in \mathbb{N}$, given an arbitrary C in G_n, A decides its satisfiability in time $t(n)$.

3.2 Main Proof

3.2.1 A Fast Satisfiability Algorithm

Transformation between different circuit types is an important building block in our proof. Yao [Yao90], Beigel and Tarui [BT94] and Allender and Gore [AG94] studied conversion from $\text{Circuit}_{\text{ACC}_m}(n^{O(1)}, O(1))$ to $\text{Circuit}_{\text{SYM}^+}(n^{\log^{O(1)} n}, \log^{O(1)} n)$. In fact, their strategy works in a more general setting.

Fix m to be an integer constant.

Theorem 49 ([BT94, AG94]) *There is a universal constant c such that for any size function $s(n)$ and any depth function $d(n)$, the family $\text{Circuit}_{\text{ACC}_m(s(n),d(n))}$ is transformable in time $2^{O((\log s(n))^{2^{cd(n)}})}$ to the family $\text{Circuit}_{\text{SYM}^+(2^{(\log s(n))^{2^{cd(n)}}}, (\log s(n))^{2^{cd(n)}})}$.*

Corollary 50 *For any small constant ϵ, any quasi-polynomial $p(n)$ and any depth function $d(n)$ of order $o(\log\log n)$, the family $\text{Circuit}_{\text{ACC}_m(p(n),d(n))}$ is transformable to the family $\text{Circuit}_{\text{SYM}^+(2^{n^\epsilon}, n^\epsilon)}$ in time $2^{O(n^\epsilon)}$.*

In [Wil10], Williams gave an algorithm for solving the satisfiability problem of SYM^+ circuits of size s over n variables in time $O((2^n + s)n^{O(1)})$. Combining it with Corollary 50, the following theorem is immediate.

Theorem 51 *There exists a constant c such that for any quasi-polynomial $p(n)$ and any depth function $d(n)$ of order $o(\log\log n)$, $\text{Circuit}_{\text{ACC}_m}(p(n), d(n))$-SAT is solvable in time $O(2^n n^c)$.*

The running time above can indeed be improved.

Theorem 52 *For any positive constant c', any quasi-polynomial $p(n)$ and any depth function $d(n)$ of order $o(\log\log n)$, Circuit$_{\mathrm{ACC}_m}(p(n), d(n))$-SAT is solvable in time $O(\frac{2^n}{n^{c'}})$.*

Proof. Let c be the constant in Theorem 51. Given an $\mathrm{ACC}_m(p(n), d(n)$ circuit over n variables, when the first $(c + c')\log n$ inputs are set to definite values, we simplify it to obtain a circuit over $n - (c + c')\log n$ many variables. Hence, by fixing the first $(c + c')\log n$ input variables to all possible sequences, we get $n^{c+c'}$ many circuits. Create a new circuit by feeding their outputs to a single OR gate. The size of this new circuit is bounded by $p(n)n^{c+c'}$ and its depth is only increased by one. Note that $p(n)n^{c+c'}$ is still a quasi-polynomial in $(n - (c + c')\log n)$, and $d(n) + 1$ is in $o(\log\log(n - (c + c')\log n))$ as well. By Theorem 51, its satisfiability can be determined in time $O(2^{n-(c+c')\log n}(n - (c + c')\log n)^c)$ which is $O(\frac{2^n}{n^{c'}})$. This finishes our arguments since the satisfiability problem for the new circuit is equivalent to the one for the original circuit. □

Note: The above strategy is very similar to the one adopted by [Wil10], where about n^δ many input variables are set in a single copy. However, it is crucial for our work to keep the size of the final circuit within quasi-polynomial (compared to $2^{n^{O(\delta)}}$ in [Wil10]) in order to apply Theorem 51.

3.2.2 Proof of Theorem 3

In this section, we present our main lower bound result via the framework invented by Williams [Wil10]. The following notions will be useful.

Definition 53 *Let $x = x_0x_1x_2...x_{|x|-1}$ be a binary string, where $|x|$ is the size of x. We say x is* succinctly represented by the circuit *C if C has $\lceil\log(|x|+1)\rceil$ many input bits and moreover, for all $0 \le i \le |x| - 1$, $C(i) = x_i$ while its output can be arbitrary otherwise. We call such a circuit C as a* succinct representation *of x.*

Let ϕ be a 3-CNF formula with n variables and m clauses. ϕ is succinctly represented by the circuit *C' if C' has $\lceil\log(m + 1)\rceil$ many input bits and furthermore, on the input $0 \le i \le$*

$m-1$, *$C'(i)$'s output is the standard binary encoding of the ith clause. Hence, C' has roughly* $3(\lceil \log(n+1) \rceil + 1)$ *output bits, the amount which is needed to encode three literals. We say that C' is a* succinct representation or compression *of ϕ.*

Theorem 54 (Theorem 3 restated)

$$\text{NEXP} \not\subseteq \text{ACC}(n^{\log^{O(1)} n}, o(\log \log n)).$$

Proof. Suppose NEXP $\subseteq$ ACC$(n^{\log^{O(1)} n}, o(\log \log n))$. The first step of our proof is to note that, because of Theorem 52, it is possible to state a slight variant of Lemma 3.1 of [Wil10].

Lemma 55 *There is a universal positive constant c with the following property. Assume that* P $\subseteq$ ACC$(n^{\log^{O(1)} n}, o(\log \log n))$*, then for every $L \in$* NTime$[2^n]$*, there is a nondeterministic algorithm $\mathcal{A}$, an integer constant m, a quasi-polynomial $p'(n)$ and a depth function $d'(n)$ of order $o(\log \log n)$ such that*

- *$\mathcal{A}$ runs in $O(\frac{2^n}{n^{c'}})$ time,*
- *for every instance x with $|x| = n$, $\mathcal{A}(x)$ either rejects or prints a circuit $C_x \in G_{n+c \log n}$ where $G_{n+c \log n} \in$ Circuit$_{\text{ACC}_m}(p'(n), d'(n))$ such that $x \in L$ if and only if C_x is the compression of a satisfiable 3-CNF formula F_x of size $2^n \cdot n^{O(1)}$, and*
- *there is at least one computation path $\mathcal{A}(x)$ that outputs C_x.*

Hence, Lemma 55 implies that as long as deciding the satisfiability of succinct 3-CNF instances such as C_x can be achieved in nondeterministic time $O(\frac{2^n}{n^{c'}})$ for any c', then NTime$[2^n] \subseteq$ NTime$[\frac{2^n}{n^{c'}}]$, in contradiction to the nondeterministic time hierarchy [Zak83]. Therefore, we are done except for showing that the satisfiability of C_x can be tested in this time bound, assuming that NEXP $\subseteq$ ACC$(n^{\log^{O(1)} n}, o(\log \log n))$.

The following theorem is a variant of Theorem 5.2 in [Wil10]. It is also implicit in the work of Impagliazzo, Kabanets and Wigderson [IKW02].

Theorem 56 ([IKW02, Wil10]) NEXP $\subseteq$ SIZE$(n^{\log^{O(1)} n})$ *implies that for every language L in* NEXP*, there exists a quasi-polynomial p such that $\forall x \in L$, there exists a witness w for x with the property that the boolean function whose truth table is given by w can be computed by a general circuit of size at most $p(|x|)$.*

In other words, every instance in L has a succinctly represented witness. In particular, every compressed 3*-CNF formula has a succinct satisfying assignment since the Succinct SAT Problem is in* NEXP.

Our assumption that NEXP $\subseteq$ ACC($n^{\log^{O(1)} n}, o(\log\log n)$) implies NEXP $\subseteq$ SIZE($n^{\log^{O(1)} n}$), so obviously the conclusion in Theorem 56 holds.

Lemma 57 (Folklore) *If* P $\subseteq$ ACC($n^{\log^{O(1)} n}, o(\log\log n)$)*, then there exists a universal constant m' such that for any quasi-polynomial $p(n)$, there exists a quasi-polynomial $p'(n)$ and a depth function $d(n)$ of order $o(\log\log n)$ such that Circuit$_{General}(p(n))$ is transformable to Circuit$_{\text{ACC}_{m'}}(p'(n), d(n))$.*

Proof. The Circuit Value Problem (CVP) is in P, and hence, there exists an integer constant m', a quasi-polynomial $q(n)$ and a depth function $d'(n) = o(\log\log n)$ such that CVP is computed by a family of ACC$_{m'}$ circuits of size at most $q(n)$ and depth bounded by $d'(n)$. Under the standard encoding of circuits, this implies that any general circuit of size at most $p(n)$ has an equivalent ACC$_{m'}$ circuit of size at most $q(p^2(n))$ and depth bounded by $d'(p^2(n))$. Since $q(p^2(n))$ is still a quasi-polynomial in n and $d'(p^2(n)) = o(\log\log n)$, our claim holds. □

Theorem 56 tells us that for every x in L, there exists a witness w that is succinctly represented by a circuit of quasi-polynomial size. By Lemma 57, this circuit can be assumed to be a quasi-polynomial-size ACC$'_m$ circuit C_w of depth $o(\log\log n)$. Thus analogous to the work of Williams [Wil10], our algorithm for deciding the satisfiability of the succinctly represented 3-CNF instance C_x proceeds as the following steps.

1. Guess the circuit C_w of quasi-polynomial size and depth $o(\log\log n)$, where w is a witness for C_x being satisfiable.

2. Build a circuit C of the following form: On input i, use C_x to obtain the encoding of the ith clause of the formula F_x. Querying C_w, find the values of the three variables occurring in this clause, according to the witness w.

3. C rejects if and only if these values cause cause the clause to evaluate to 1.

Note that C is unsatisfiable if and only if every clause of F_x is satisfied by w.

Fact 58 *For two fixed integers m and m', there exists a polynomial r such that any* ACC *circuit containing both* MOD_m *and* $\text{MOD}_{m'}$ *gates of size at most s can be simulated uniformly by an* ACC_l *circuit of the same depth and size at most $r(s)$ where $l = m \cdot m'$.*

By fact 58, C is a quasi-poly-size ACC_l circuit of depth at most $d(n) + d'(n) + O(1)$ and by Theorem 52, its satisfiability is decidable in time $O(\frac{2^n}{n^{c'}})$ for any c', which concludes our proof for the main theorem. □

3.3 Discussions

We have not fully exploited the strength of the machinery behind Theorem 49. The original form of the transformation provides a large set of parameters which can be tuned smoothly. For instance, one can allow $m(n) = \{l_1, l_2, ...\}$ to be a slowly growing (say, of order $O(\log \log n)$) integer sequence rather than a fixed constant and consider the circuit families $\text{ACC}_{m(n)}$ where the nth circuit contains the presence of MOD_{l_i} gates for all $i \leq n$. It is easy for the readers who are familiar with the framework of [Yao90], [BT94] and [AG94] to verify that NEXP does not have non-uniform $\text{ACC}_{m(n)}$ circuits of quasi-polynomial size and non-constant depth. This phenomenon has been observed by several authors, [Bar92], [BTT92] etc. And their further investigations made it explicit that $\text{SYM}^+(n^{\log^{O(1)} n}, \log^{O(1)} n)$ actually encompasses a circuit complexity class presumably larger than $\text{ACC}(n^{O(1)}, O(1))$, where every $\text{ACC}(n^{O(1)}, O(1))$ circuit has an extra symmetric gate at the top. Hence, it is natural to conjecture that NEXP is not contained in this class either. However, the proof of Theorem 52 introduces too many duplicate symmetric gates, which falls beyond the reach of current techniques. Note that Beigel [Bei94] showed that polylog majority gates can be merged into one at the top, but his results would yield the trivial bound 2^{n^c} for some $c > 1$ in our case.

We would like to draw the comparison between this work and [Smo87]. The main technical difficulty which prevents us from obtaining a depth lower bound of order $\Omega(\frac{\log n}{\log \log n})$ is that each application of modulus-amplifying polynomials creates extra AND gates of large fan-in. This in turn causes the snowball effect of the blow-up in the final circuit size. Thus, new ideas are needed in order to improve the current depth lower bound.

Chapter 4

On the Power of $\text{IMM}_{2,n}$

4.1 Preliminaries

Let the underlying field be $\mathbb{F}$. Let $q(\overline{x}) \in \mathbb{F}[\overline{x}]$ be a multivariate polynomial over a set of variables $\overline{x}$. A projection p on $q(\overline{x})$ is an operation to generate new polynomials; a projection is described by a set of assignments $\{x_i \leftarrow v_i\}$, where the values v_i come from a particular set (to be specified later), and each variable $x_i \in \overline{x}$ appears at most once on the left-hand-side of a rule in p; furthermore, variables on the left-hand-side never occur on the right-hand-side. We get the new instance $q(\overline{x})|_p$ by replacing all occurrences of x_i in $q(\overline{x})$ with its counterpart v_i and leaving untouched those variables that are not in p. We may simplify $q(\overline{x})|_p$ according to the commutative polynomial ring algebra. In this way, we say that $q(\overline{x})|_p$ *is obtained from* $q(\overline{x})$ *under the projection* p.

Let $\mathbb{H}$ be the set of homogeneous linear terms $\{c \cdot x_i \mid c \in \mathbb{F}^*, i \in \mathbb{N}\}$ where $\mathbb{F}^*$ is the set of units (i.e., non-zero elements). Let $\mathbb{S}$ be the set of simple linear terms $\{c \cdot x_i + w \mid c \in \mathbb{F}^*, i \in \mathbb{N}, w \in \mathbb{F}\}$ and $\mathbb{L}$ be the set of general linear terms $\{\sum_{i=1}^{n} c_i \cdot x_i + w \mid n \in \mathbb{N}, c_i, w \in \mathbb{F}\}$. We define a projection $p = \{x_i \leftarrow v_i\}$ to be a *homogeneous projection* if $\forall i$, $v_i \in \mathbb{H} \cup \mathbb{F}$. It is a *simple projection* if $\forall i$, $v_i \in \mathbb{S} \cup \mathbb{F}$. If $\forall i$, $v_i \in \mathbb{L}$, then p is a *regular projection*. We mention that the most restrictive of these three types of projections, homogeneous projections, are the usual types of projections studied in algebraic complexity [Val79a, BOC92].

Consider n square matrices of dimension two $m_1, m_2, \ldots, m_n$, the entries of which are distinct variables. The $(1,1)$-entry of their product $\prod_{i=1}^{n} m_i$ is a multi-linear polynomial, denoted as $\text{IMM}_{2,n}$, which is called the *nth iterated matrix multiplication polynomial of dimension two*. The matrix $m_i|_p$ is obtained from m_i under the projection p, which means that the entries of m_i are substituted by the corresponding values in p. Given a polynomial $f(\overline{x})$, it is easy to see that $f(\overline{x})$ is obtained from $\text{IMM}_{2,n}$ under some projection p if and only if $f(\overline{x})$ is

the $(1,1)$-entry of $\prod_{i=1}^{n} m_i|_p$, and moreover, the variables appearing in $m_i|_p$ belong to the set $\{x_j \mid x_j \text{ occurs in } f(\overline{x})\}$. Note that $f(\overline{x})$ is computable by some algebraic branching program of width two if and only if there exists $n \in \mathbb{N}$ such that $f(\overline{x})$ can be obtained from $\text{IMM}_{2,n}$ under regular projections.

Let M be a set of square matrices of dimension two. We say a polynomial $f(\overline{x})$ is *computable by* M if there is an integer n and a projection p such that $f(\overline{x}) = \text{IMM}_{2,n}|_p$ and furthermore, $\forall i \leq n,\ m_i|_p \in M$. In other words, $f(\overline{x})$ can be computed by the product of matrices in M.

Let $\mathbb{H}_{2\times 2}$ denote the set of square matrices of dimension two with entries from $\mathbb{H} \cup \mathbb{F}$. Similarly, let $\mathbb{S}_{2\times 2}$ ($\mathbb{R}_{2\times 2}$, respectively) denote the set of square matrices of dimension two with entries from $\mathbb{S} \cup \mathbb{F}$ ($\mathbb{L}$, respectively). Obviously, $\mathbb{H}_{2\times 2} \subseteq \mathbb{S}_{2\times 2} \subseteq \mathbb{R}_{2\times 2}$.

We divide all square matrices of dimension two whose entries belong to $\mathbb{L}$ into three groups, Indg, Idg and Pdg. The matrices in Indg are called *inherently non-degenerate matrices* and their determinants evaluate to a fixed element in $\mathbb{F}^*$ while Idg consists of *inherently degenerate matrices* with zero determinants. $\text{Pdg} = \mathbb{R}_{2\times 2} \setminus (\text{Indg} \cup \text{Idg})$ is the set of *potentially degenerate matrices*. Obviously the determinants of matrices in Pdg are nonzero polynomials of degree at least one.

Our results deal with some simple degree-two polynomials; the following facts are easy to verify.

Fact 59 *Over any field $\mathbb{F}$, $x_1x_2 + x_3x_4$ is an irreducible polynomial.*

Fact 60 *Let $\mathbb{F}$ be any field, $k \geq 2$ and let $l(\overline{x})$ be an arbitrary linear function. Then $\Sigma_{i=1}^{k} x_{2i-1}x_{2i} + l(\overline{x})$ is an irreducible polynomial, and furthermore, its degree-two homogeneous part is irreducible as well.*

We group the variables x_{2i-1} and x_{2i} together, and call each the other's *partner variable*.

Definition 61 *In a regular projection p given by $\{x_j \leftarrow v_j\}$, the partner variables x_{2i-1}, x_{2i} are called* matched *if*

- *Both of x_{2i-1} and x_{2i} appear on the left-hand-side.*

- $\{v_{2i-1}, v_{2i}\} \cap \mathbb{F} \neq \emptyset$.

It is convenient to consider a restricted class of projections:

Definition 62 *A regular projection p given by $\{x_i \leftarrow v_i\}$ is* well-formed *if every left-hand-side variable x_i is matched.*

We will make use of the fact that any projection can be "extended" to obtain a well-formed projection. However, we must first be precise about what it means for one projection to be an "extension" of another. (To see what the issue is, consider the projection $\{x_1 \leftarrow x_3, x_2 \leftarrow x_4\}$. The partner variables x_1 and x_2 are not matched, since neither of them is assigned a field element. Thus we need to consider how to "extend" projections, by not only adding new rules, but also by changing existing rules appropriately.

Definition 63 *A regular projection p is an* extension *of a projection p' if there is a projection p'' such that $p = p' \circ p''$.*

Thus to continue the example above, the projection $p' = \{x_1 \leftarrow x_3, x_2 \leftarrow x_4\}$ can be extended by $p'' = \{x_4 \leftarrow 0\}$ to obtain the projection $p = \{x_1 \leftarrow x_3, x_2 \leftarrow 0, x_4 \leftarrow 0\}$ (which is still not well-formed).

Proposition 64 *Any regular projection of size k with l unmatched left-hand-side variables can be extended to a well-formed regular projection of size at most $k+l$. Thus any regular projection of size k can be extended to a well-formed regular projection of size at most $2k$.*

Proof. The proof proceeds by induction on l. The basis, when $l = 0$, is trivial.

Now consider a regular projection p with l unmatched left-hand-side variables, where we inductively assume that any regular projection of size k' with $l' < l$ unmatched variables can be extended to a well-formed regular projection of size at most $k' + l'$. There are two cases:

Case 1: If there is an unmatched variable x whose partner variable y does not appear on the left-hand-side of any rule, then then we simply add the rule $y \leftarrow 0$. (If y appeared on the *right*-hand-side of any rule, then any such rule must also be simplified by setting y to zero. Such changes do not increase the size of the projection.) This yields a projection p' with at

most $l-1$ unmatched variables, where we have added one rule. (It is possible that there will be *fewer* than $l-1$ unmatched variables, if there were some unmatched variable z such that $z \leftarrow c \cdot y$ was a rule.) Now the claim follows by induction.

Case 2: If Case 1 does not hold, then there must be a pair of unmatched variables that are partners (without loss of generality call them x_1 and x_2) such that the projection has rules $x_1 \to v_1$ and $x_2 \to v_2$, where $\{v_1, v_2\} \cap \mathbb{F} = \emptyset$. Since p is a regular projection, v_1 is of the form $c_0 + \sum_{k=1}^{n} c_k y_k$, where none of the variables y_i appear on the left-hand-side of any rule in p. Note that the rule $y_1 \leftarrow (-1/c_1) \cdot (c_0 + \sum_{k=1}^{n} c_k y_k)$ has the effect of setting x_1 to 0. Let z be the partner variable of y_1; note that z does not appear on the left-hand side of any rule (because otherwise Case 1 would have applied). Thus removing the rule $x_1 \to v_1$ and adding the rules $x_1 \leftarrow 0, y_1 \leftarrow -1/c_1 \cdot (c_0 + \sum_{k=1}^{n} c_k y_k), z \leftarrow 0$ has at most $l-2$ unmatched variables (since x_1 and x_2 are now both matched, as are y_1 and z), and it has two more rules than p. As above, it is now necessary to simplify any rule in which y_1 or z appeared on the right-hand-side, but this does not increase the size of the projection. Now the claim follows by induction. □

The definition of "well-formed projection" is designed to make the following proposition obvious:

Proposition 65 *Let $k, n \in \mathbb{N}$, with $n - k \geq 2$. Consider the polynomial $\Sigma_{i=1}^{n} x_{2i-1}x_{2i}$. Then under any well-formed regular projection p of size $2k$, $f(\overline{x})|_p = \Sigma_{i=1}^{n-k} x_{2i-1}x_{2i} + l(\overline{x})$ (up to re-numbering the variables) is also an irreducible polynomial, where $l(\overline{x})$ is a linear function. Furthermore, its degree-two homogeneous part is also irreducible.*

In this work, we will show that certain constant-size polynomials are not computable by various families of matrices over any algebraically closed field. By the following fact, we may as well assume that the underlying field $\mathbb{F}$ is algebraically closed.

Fact 66 *Let $\mathbb{F}'$ be the algebraic closure of $\mathbb{F}$ and let M be a set of matrices. For any polynomial $f(\overline{x})$, if $f(\overline{x})$ is computable by M over $\mathbb{F}$, then it is computable by M over $\mathbb{F}'$ as well.*

4.2 $\text{IMM}_{2,n}$ under homogeneous projections

In this section, we will show that the computational power of the family $\{\text{IMM}_{2,n} \mid n \in \mathbb{N}\}$ under homogeneous projections is very limited.

Recall that $\mathbb{H}_{2\times 2}$ denotes the set of square matrices of dimension two with entries from $\mathbb{H} \cup \mathbb{F}$. We will show that it causes no loss of computational power, if we restrict the type of matrices that are used in $\mathbb{H}_{2\times 2}$ computations. First, however, it is very useful to observe that $\mathbb{H}_{2\times 2}$ computations correspond exactly to a type of straight-line programs.

Let μ be a set of allowable straight-line program instructions (rules), and let R_i^t denote the contents of the register R_i at time t. A straight-line program P over the rule set μ using 2 registers (μ-SLP) is a sequence of pairs of instructions from μ, denoted as $\{(s_1^t, s_2^t) \mid 1 \leq t \leq |P|, (s_1^t, s_2^t) \in \mu\}$, where $|P|$ is the size of the program. P computes a function $p(\overline{x})$ in the natural way: Initially, $R_1^0 = 1$ and $R_2^0 = 0$. At the t-th step, R_i is updated according to the rule s_i^t. The final output $p(\overline{x})$ is stored as $R_1^{|P|}$. In this section, we consider only instructions that come from the set $\mu_{\mathbb{H}_{2\times 2}} = \{(R_1^{t+1} \leftarrow a{\cdot}R_1^t + b{\cdot}R_2^t,\ R_2^{t+1} \leftarrow a'{\cdot}R_1^t + b'{\cdot}R_2^t) \mid a, b, a', b' \in \mathbb{H} \cup \mathbb{F},\ t \in \mathbb{N}\}$. Under these assumptions, each R_i^t is a polynomial over the variables $\{x_j \mid j \in \mathbb{N}\}$. It is not hard to see that $\mu_{\mathbb{H}_{2\times 2}}$-SLPs and $\text{IMM}_{2,n}$ under homogeneous projections compute the same set of polynomials. (Similar observations were made by Ben-Or and Cleve [BOC92].) Furthermore, for any subset $N \subseteq \mathbb{H}_{2\times 2}$, there is a corresponding rule set $\mu_N \subseteq \mu_{\mathbb{H}_{2\times 2}}$ such that a polynomial $f(\overline{x})$ is computable by N if and only if there is a μ_N-SLP for it. Hence, given an arbitrary μ_N-SLP P, we may abuse the notations and identify the ith pair of instructions with its matrix representations m_P^i, which means that P can also be characterized by a sequence of matrices $\{m_P^i \mid 1 \leq i \leq |P|\}$.

4.2.1 Classification of $\mathbb{H}_{2\times 2} \cap$ Indg

Now, we present a collection μ_N of rules (corresponding to a subset N of matrices in $\mathbb{H}_{2\times 2}$) which we claim suffice to simulate any straight-line program using the rules $\mu_{\mathbb{H}_{2\times 2} \cap \text{Indg}}$. Let $a, b, c, d \in \mathbb{F}^*$.

1. Transposition rule.

$$\begin{aligned} R_1^{t+1} &\leftarrow R_2^t \\ R_2^{t+1} &\leftarrow R_1^t \end{aligned} \quad \text{given by matrix} \quad \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

2. Scalar rules.

$$\begin{aligned} R_1^{t+1} &\leftarrow a \cdot R_1^t \\ R_2^{t+1} &\leftarrow b \cdot R_2^t \end{aligned} \quad \text{given by matrix} \quad \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}$$

3. Offsetting rules of degree one.

   (a)

$$\begin{aligned} R_1^{t+1} &\leftarrow a \cdot R_1^t + c \cdot x_i \cdot R_2^t \\ R_2^{t+1} &\leftarrow b \cdot R_2^t \end{aligned} \quad \text{given by matrix} \quad \begin{bmatrix} a & c \cdot x_i \\ 0 & b \end{bmatrix}$$

   (b)

$$\begin{aligned} R_1^{t+1} &\leftarrow a \cdot R_1^t \\ R_2^{t+1} &\leftarrow c \cdot x_i \cdot R_1^t + b \cdot R_2^t \end{aligned} \quad \text{given by matrix} \quad \begin{bmatrix} a & 0 \\ c \cdot x_i & b \end{bmatrix}$$

4. Offsetting rules of degree zero.

   (a)

$$\begin{aligned} R_1^{t+1} &\leftarrow a \cdot R_1^t + c \cdot R_2^t \\ R_2^{t+1} &\leftarrow b \cdot R_2^t \end{aligned} \quad \text{given by matrix} \quad \begin{bmatrix} a & c \\ 0 & b \end{bmatrix}$$

   (b)

$$\begin{aligned} R_1^{t+1} &\leftarrow a \cdot R_1^t \\ R_2^{t+1} &\leftarrow c \cdot R_1^t + b \cdot R_2^t \end{aligned} \quad \text{given by matrix} \quad \begin{bmatrix} a & 0 \\ c & b \end{bmatrix}$$

5. Other non-degenerate linear transformations.

$$\begin{aligned} R_1^{t+1} &\leftarrow a \cdot R_1^t + c \cdot R_2^t \\ R_2^{t+1} &\leftarrow d \cdot R_1^t + b \cdot R_2^t \end{aligned} \quad \text{given by matrix} \quad \begin{bmatrix} a & c \\ d & b \end{bmatrix}$$

   where $ab - cd \neq 0$.

Observation 67 *Any straight-line program using $\mu_{\mathbb{H}_{2\times 2} \cap \mathit{Indg}}$ can be simulated by a straight-line program using μ_N. That is, without loss of generality, one can assume that any straight-line program P has the following properties.*

- *If the transposition matrix is ever adopted by P, it is applied only once as the final pair of instructions. (This is because we can cancel adjacent transpositions, and shift any single transposition toward the end of the program via the following transformation:*

$$\begin{bmatrix} v & u \\ z & y \end{bmatrix} = \begin{bmatrix} u & v \\ y & z \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

$$\begin{bmatrix} y & z \\ u & v \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} u & v \\ y & z \end{bmatrix}$$

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} u & v \\ y & z \end{bmatrix} = \begin{bmatrix} z & y \\ v & u \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

- *The following matrices need never appear, because they are transpositions of rules in μ_N (and transpositions introduced in this way can be eliminated).*

$$\begin{bmatrix} 0 & a \\ b & 0 \end{bmatrix}, \begin{bmatrix} 0 & a \\ b & c \end{bmatrix}, \begin{bmatrix} c & a \\ b & 0 \end{bmatrix}, \begin{bmatrix} 0 & a \\ b & c \cdot x_i \end{bmatrix}, \begin{bmatrix} c \cdot x_i & a \\ b & 0 \end{bmatrix}$$

- *This leaves only the rule set μ_N.*

4.2.2 Structure of $\mu_{\mathbb{H}_{2\times 2}\cap}$**Indg**-SLPs and its implications

Definition 68 *Let $deg(f)$ denote the degree of the polynomial f. For any straight-line program P, let $deg(P,t) = deg(R_1^t) + deg(R_2^t)$ be* the degree of P at time t. *We call the sequence of natural numbers $deg(P,0), deg(P,1), \ldots, deg(P,|P|)$* the degree sequence of P.

An ordered pair of non-negative integers (t_1, t_2), where $t_1 + 1 < t_2$, is called a mesa *in the degree sequence of P if there exists $d > 0$ such that*

- *For all $t_1 < t' < t_2$, $deg(P,t') = d$;*
- *$deg(P,t_1) < d$;*
- *$deg(P,t_2) < d$.*

The number d is called the height *of this mesa.*

Fact 69 *The operations in Observation 67 that simplify the straight-line program P do not change the height of any mesa in which the operations are applied.*

Now we are ready to show our structural theorem for $\mu_{\mathbb{H}_{2\times 2}\cap\text{Indg}}$-SLPs.

Theorem 70 *If a polynomial f is computable by $\mathbb{H}_{2\times 2}\cap$Indg, then there is a $\mu_{\mathbb{H}_{2\times 2}\cap\text{Indg}}$-SLP P for f with the property that there are no mesas in the degree sequence of P.*

Proof. By our assumption, there is some $\mu_{\mathbb{H}_{2\times 2}\cap\text{Indg}}$-SLP P' computing f. If P' does not contain any mesas in its degree sequence, then we are done. Otherwise, we will show how to obtain P from P' by a series of transformations. At every step, we turn the current P' into an equivalent $\mu_{\mathbb{H}_{2\times 2}\cap\text{Indg}}$-SLP while reducing the total height of all mesas by at least one. Ultimately we will obtain a $\mu_{\mathbb{H}_{2\times 2}\cap\text{Indg}}$-SLP P with the desired property. Hence, it suffices to verify the correctness of a single step.

Let (t_1, t_2) be the first mesa in the current P' and d be its height. There are three cases to consider.

1. $\deg(R_1^{t_1+1}) > \deg(R_2^{t_1+1})$.

   We claim that the only instruction that can produce this outcome at time $t_1 + 1$ is the degree-one offsetting rule $3(a)$. Rule 2 is impossible since it only scales the registers by a constant factor respectively. Rule $3(b)$ implies that $\deg(R_1^{t_1+1}) = \deg(R_1^{t_1})$; there are two subcases to consider:

   - If $\deg(R_1^{t_1}) \geq \deg(R_2^{t_1})$, then $\deg(R_1^{t_1+1}) \leq \deg(R_2^{t_1+1})$, a contradiction to our assumption that $\deg(R_1^{t_1+1}) > \deg(R_2^{t_1+1})$.
   - If $\deg(R_1^{t_1}) < \deg(R_2^{t_1})$, then $\deg(R_2^{t_1+1}) \leq \deg(R_2^{t_1})$. This contradicts our assumption that $\deg(P, t_1) < \deg(P, t_1 + 1)$.

   For similar reasons, one can show that rules $4(a)$ and $4(b)$ are not applicable either. There are two cases that arise, in dealing with rule 5:

   - If $\deg(R_1^{t_1}) \neq \deg(R_2^{t_1})$, then under rule 5, $\deg(R_1^{t_1+1}) = \deg(R_2^{t_1+1})$, which contradicts our assumption that $\deg(R_1^{t_1+1}) > \deg(R_2^{t_1+1})$;

- If $\deg(R_1^{t_1}) = \deg(R_2^{t_1})$, then $\deg(P', t_1) \geq \deg(P', t_1 + 1)$, which contradicts our assumption that (t_1, t_2) is a mesa.

$\forall t_1 < t' < t_2,\ \deg(P', t') = d$ and $\deg(P', t_2) < d$ implies that rules $3(b)$, $4(b)$ and 5 are impossible at time t' (and at time t_2), since under our assumptions they would increase the degree of R_2 while maintaining the degree of R_1. Hence, for all $t_1 < t' \leq t_2$, the product $\prod_{i=t_1+1}^{t'} m_{P'}^i$ is an upper triangular matrix of the form $\begin{bmatrix} a & g_{t'} + w \\ 0 & b \end{bmatrix}$, where $w \in \mathbb{F}$, $a, b \in \mathbb{F}^*$ and $g_{t'}$ is a linear homogeneous polynomial. In other words, $R_1^{t'} = a \cdot R_1^{t_1} + (g_{t'} + w) \cdot R_2^{t_1}$ and $R_2^{t'} = b \cdot R_2^{t_1}$. Since $\deg(P', t_2) < d = \deg(P', t_1 + 1)$, it follows that $\deg(R_1^{t_1}) < \deg(R_1^{t_1+1})$ and $g_{t_2} = 0$. Thus, we can replace the whole computation between t_1 and t_2 by a simple application of rule 2 or $4(a)$ while avoiding the mesa (t_1, t_2).

2. $\deg(R_1^{t_1+1}) < \deg(R_2^{t_1+1})$.

   This is completely analogous to case 1.

3. $\deg(R_1^{t_1+1}) = \deg(R_2^{t_1+1})$.

   We argue that neither of rules $3(a)$ and $3(b)$ can happen at time $t_1 + 1$. We study the reasons for $3(a)$ and those for $3(b)$ are symmetric.

   - If $\deg(R_1^{t_1}) \leq \deg(R_2^{t_1})$, then $\deg(R_1^{t_1+1}) > \deg(R_2^{t_1+1})$, a contradiction to our assumption $\deg(R_1^{t_1+1}) = \deg(R_2^{t_1+1})$.
   - If $\deg(R_1^{t_1}) > \deg(R_2^{t_1})$, then $\deg(P', t_1 + 1) < \deg(P', t_1)$ since $\deg(R_2^{t_1}) = \deg(R_2^{t_1+1})$. This contradicts our assumption that (t_1, t_2) is a mesa.

   Furthermore, $\forall t_1 < t' < t_2,\ \deg(P', t') = d$ implies that rules $3(a)$ and $3(b)$ are impossible at time t' (and at time t_2). Thus, we obtain that for all $t_1 < t' \leq t_2$, the product $\prod_{i=t_1+1}^{t'} m_{P'}^i$ is a non-degenerate linear transformation, which can be captured by one of the other rules or their transposed counterparts. The analysis of this case can now be completed similarly to Case 1, by appealing to Observation 67 and Fact 69.

In all cases, we are able to reduce the total height of all mesas in P' by at least one, which concludes our proof. □

Corollary 71 *If a polynomial $f(\overline{x})$ is computable by $\mathbb{H}_{2\times 2}\cap$Indg, then there is a $\mu_{\mathbb{H}_{2\times 2}\cap \textit{Indg}}$-SLP for $f(\overline{x})$ with a monotonically nondecreasing degree sequence.*

The analysis in the proof of Theorem 70 also allows one to draw the following conclusions:

Fact 72 *For any $\mu_{\mathbb{H}_{2\times 2}\cap \textit{Indg}}$-SLP P, $\forall 0 < t \leq |P|$, if $deg(P,t) > deg(P,t-1)$, then only one of the following two scenarios can happen.*

- *If either $3(a)$ or $3(b)$ is applied at time t, then $|deg(R_1^t) - deg(R_2^t)| = 1$.*
- *If the other rules are used at time t, then $deg(R_1^t) = deg(R_2^t)$.*

Lemma 73 *If P is a $\mu_{\mathbb{H}_{2\times 2}\cap \textit{Indg}}$-SLP with a monotonically nondecreasing degree sequence, then for all $0 \leq t \leq |P|$, $|deg(R_1^t) - deg(R_2^t)| \leq 1$. Furthermore, we can assume that $deg(R_1^{|P|}) \geq deg(R_2^{|P|})$.*

Proof. The first part follows naturally from Fact 72. It suffices to justify the second claim. Suppose there is no presence of rule 1 in P. Let $r = \deg(R_1^{|P|})$. If $\deg(R_2^{|P|}) = r+1$, then from the time t' when $\deg(R_2^{|P|})$ is increased to $r+1$, the polynomial in R_1 will change only by multiplication by a nonzero field element. This is because of the monotonicity of the degree sequence. Hence, we can skip the steps following time t' and substitute them by an appropriate scalar matrix instead. Then the degree of R_2 will remain no more than r in the new $\mu_{\int}$-SLP. The argument in the case when rule 1 appears is completely symmetric. □

Theorem 74 *Let $f(\overline{x})$ be a polynomial of total degree at least two whose highest-degree homogeneous part is irreducible. Then $f(\overline{x})$ is not computable by $\mathbb{H}_{2\times 2} \cap$ Indg.*

Proof. The proof is by contradiction. Suppose there exists a $\mu_{\mathbb{H}_{2\times 2}\cap \text{Indg}}$-SLP P for $f(\overline{x})$. By Corollary 71 and Lemma 73, P has a monotonically nondecreasing degree sequence, and $\deg(R_1^{|P|}) = \deg(f)$ while $\deg(R_2^{|P|}) \leq \deg(f)$. If $\deg(R_2^{|P|}) < \deg(f)$, the analysis of Theorem 70 and Fact 72 reveals that the highest-degree homogeneous part of $f(\overline{x})$ contains a linear factor and hence, is reducible, which is a contradiction to our assumption. So assume $\deg(R_2^{|P|}) = \deg(f)$ and without loss of generality, assume that R_2's degree reaches $\deg(f)$ first, at some time t_0, and that R_1's degree is raised to $\deg(f)$ at some time $t > t_0$. Thus all

of the highest-degree monomials in $R_2^{t_0}$ come from $c \cdot x_i \cdot R_1^{t_0-1}$. For any polynomial g, let $H_d(g)$ denote the degree-d homogeneous part of polynomial g. An easy induction shows that, for all t' such that $t_0 \leq t' \leq t$, $H_{\deg(f)-1}(R_1^{t_0-1})$ divides $H_{\deg(f)}(R_2^{t'})$. By Fact 72, rules $3(a)$ and $3(b)$ can not be applied at time t. Since the degree sequence is stable from then on, they will not happen afterwards either. Hence, by the linearity of the remaining rules, we claim that $\forall t \leq t' \leq |P|,\ \exists a, b \in \mathbb{F}^*,\ H_{\deg(f)}(R_1^{t'}) = aH_{\deg(f)}(R_2^{t'}) = bH_{\deg(f)}(R_2^{t-1})$, which we argued above is divided by $H_{\deg(f)-1}(R_1^{t_0-1})$. This is a contradiction to our assumption that it is irreducible. This concludes our proof. □

Remark 75 *The proof of Theorem 74 reveals that if $f(\overline{x})$ is computable by $\mathbb{H}_{2\times 2} \cap$ Indg, then the highest-degree homogeneous part of $f(\overline{x})$ can be completely factored into homogeneous linear polynomials.*

4.2.3 Limitation of $\mu_{\mathbb{H}_{2\times 2}}$-SLPs

First, by the following lemma, we can assume without loss of generality that for any $\mu_{\mathbb{H}_{2\times 2}}$-SLP P and any inherently degenerate matrix m in P, the entries of m all belong to $\mathbb{F}$.

Proposition 76 *If a matrix $m \in \mathbb{H}_{2\times 2} \cap$ Idg contains at least one entry from $\mathbb{H}$, then m can be factored into a product of matrices, exactly one of which, denoted as m_1, belongs to Idg and furthermore, all m_1's entries are from $\mathbb{F}$.*

Proof. If m has a zero column, then without loss of generality, m is either $\begin{bmatrix} a \cdot x_i & 0 \\ w & 0 \end{bmatrix}$ or $\begin{bmatrix} a \cdot x_i & 0 \\ b \cdot x_j & 0 \end{bmatrix}$ where $a, b \in \mathbb{F}^*$ and $w \in \mathbb{F}$. $\begin{bmatrix} a \cdot x_i & 0 \\ w & 0 \end{bmatrix} = \begin{bmatrix} x_i & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} a & 0 \\ w & 0 \end{bmatrix}$ while $\begin{bmatrix} a \cdot x_i & 0 \\ b \cdot x_j & 0 \end{bmatrix} = \begin{bmatrix} x_i & 0 \\ 0 & x_j \end{bmatrix} \begin{bmatrix} a & 0 \\ b & 0 \end{bmatrix}$. In both cases, we obtain the desired factorization for m. The case where m has a zero row is symmetric.

For the other cases, it is not hard to see that under our assumption, m can be turned into a matrix with either a zero column or a zero row via multiplication by a non-degenerate linear transformation. Our proof is completed by referring to the previous case analysis. □

Note: The statement of Proposition 76 can be generalized for matrices in $\mathbb{S}_{2\times 2} \cap$ Idg and $\mathbb{R}_{2\times 2} \cap$Idg with almost the same proof. Hence, we can assume that for any $\mu_{\mathbb{S}_{2\times 2}}$-SLP ($\mu_{\mathbb{R}_{2\times 2}}$-SLP, respectively) P and any inherently degenerate matrix m in P, the entries of m all belong to $\mathbb{F}$.

Lemma 77 *If a nonzero polynomial $f(\overline{x})$ is computable by a straight-line program P, then P does not contain any matrix of the form* $\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$.

Proof. Suppose P does have at least one such matrix, then the product of matrices in P evaluates to $\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$, a contradiction to our assumption about $f(\overline{x})$. □

Given a projection p and a straight-line program $P = \{m_P^i \mid 1 \leq i \leq |P|\}$ computing a polynomial $f(\overline{x})$, we obtain the straight-line program $P|_p = \{m_P^i|_p \mid 1 \leq i \leq |P|\}$, which is a new straight-line program (not incorporating any simplifications). Moreover, $P|_p$ computes $f(\overline{x})|_p$. Note that this definition applies for any type of projections. In the remaining part, by Propositions 64 and 65, the polynomial $f(\overline{x})$ considered will be nonzero under any regular projection of size at most four, which leads to the following lemma.

Lemma 78 *There does not exist a matrix in P such that all of its entries belong to $\mathbb{H}$. This implies all matrices in P must contain an entry from $\mathbb{F}$.*

Proof. Suppose P does have one such matrix, and without loss of generality, assume that it has the form $\begin{bmatrix} c_1x_1 & c_2x_2 \\ c_3x_3 & c_4x_4 \end{bmatrix}$, where $\forall 1 \leq i \leq 4, c_i \in \mathbb{F}^*$ and the x_i's need not be distinct. Consider the projection $p = \{x_i \leftarrow 0 \mid 1 \leq i \leq 4\}$. Then $f(\overline{x})|_p$ is nonzero while $P|_p$ contains $\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$, in contradiction to Lemma 77. □

Lemma 79 *For any matrix $m \in P$ which belongs to $\mathbb{H}_{2\times 2} \cap$ Pdg, there exists a homogeneous projection p of size at most three such that $m|_p$ is degenerate and all of the entries in $m|_p$ belong to $\mathbb{F}$. Moreover, there is a well-formed homogeneous projection q of size at most six extending p.*

Proof. The determinant of m, denoted as $\det(m)$, is a polynomial of degree at least one. Since $\mathbb{F}$ is algebraically closed, the variety of $\det(m)$ is always non-empty and furthermore, by Lemma 78, $\det(m)$ contains at most three variables. This provides us the projection promised in the first claim. Proposition 64 implies the correctness of the second claim. □

Definition 80 *We call such a well-formed homogeneous projection q as in Lemma 79 a* degenerating projection *for the potentially degenerate matrix m.*

Lemma 81 *Let $f(\overline{x})$ be a polynomial and P be one of its $\mu_{\mathbb{H}_{2\times 2}}$-SLPs. Suppose that there exists $0 < t \leq |P|$ such that m_P^t is a potentially degenerate matrix. Let p be one of its degenerating projections. Let $P' = P|_p$ and let $R_i^t(P')$ be the contents of R_i at time t in P'. Then up to the permutation of the indices, only one of the three following cases will happen.*

1. *$R_1^t(P') = R_2^t(P') = 0$ and $f(\overline{x})|_p = 0$. This is the uninteresting case and we ignore it in the remainder of the proof.*
2. *$R_1^t(P') \in \mathbb{F}^*$ and $R_2^t(P') = w \cdot R_1^t(P')$ for some $w \in \mathbb{F}$.*
3. *$R_1^t(P')$ is a polynomial of degree at least one, $R_2^t(P') = w \cdot R_1^t(P')$ for some $w \in \mathbb{F}$, and $f(\overline{x})|_p$ is divisible by $R_1^t(P')$.*

Proof. Our assumption is that $m_{P'}^t$ is a degenerate matrix; let us say that it is $\begin{bmatrix} a & c \\ d & b \end{bmatrix}$, where $ab - cd$=0. Assume for now that $c \neq 0$. (The case where $c = 0$ is easier.) Let f and g be the polynomials given by $R_1^{t-1}(P')$ and $R_2^{t-1}(P')$, respectively. Thus $R_1^t(P') = af + cg$ and $R_2^t(P') = (b/c)(af + cg)$. Thus $R_2^t(P')$ is a multiple of $R_1^t(P')$, and an easy induction shows that $R_1^t(P')$ will stay as a common factor of both registers from that point on (and thus $R_1^t(P')$ also divides $f(\overline{x})|_p$. □

Corollary 82 *If $f(\overline{x})|_p$ is a nonzero irreducible polynomial and the other hypotheses of Lemma 81 hold, then $R_1^t(P') = c \cdot f(\overline{x})|_p$ for some $c \in \mathbb{F}^*$.*

Definition 83 *Let $f(\overline{x})$, P, m_P^t and P' satisfy the conditions of Lemma 81. If under the degenerating projection p, case 2 of Lemma 81 happens, then we call p a* cutting projection for m_P^t in P. *If instead we have case 3, then we call p a* finishing projection for m_P^t in P.

Observation 84 *Let $f(\overline{x})$ be a polynomial such that under any well-formed homogeneous projection q of size at most six, $f(\overline{x})|_q$ is always a nonzero irreducible polynomial. Let P be a $\mu_{\mathbb{H}_{2\times 2}}$-SLP for $f(\overline{x})$, and let m_P^t, p and P' be the corresponding objects as in Lemma 81. We will show how to obtain a $\mu_{\mathbb{H}_{2\times 2}}$-SLP for $f(\overline{x})|_p$ from P as follows:*

- *If the projection p is a cutting projection for m_P^t in P, then we can simply ignore the instructions in P' before time t (including the t-th instruction), and concatenate a single instruction, which is a linear transformation from the initial condition $(R_1^0, R_2^0) = (1, 0)$ to the current status $(R_1^t(P'), R_2^t(P'))$, with the remaining segment of P'. This produces a $\mu_{\mathbb{H}_{2\times 2}}$-SLP of size at most $|P| - t + 1$ for $f(\overline{x})|_p$.*

- *If p is a finishing projection for m_P^t in P, then by Corollary 82, $R_1^t(P')$ is a nonzero multiple of $f(\overline{x})|_p$ and moreover, $R_1^t(P') = a \cdot R_1^{t-1}(P') + b \cdot R_2^{t-1}(P')$, where $a, b \in \mathbb{H} \cup \mathbb{F}$, since by Lemma 79, all of entries in $m_P^t|_p$ are field elements. We claim that one of a and b must be a unit. Otherwise, $f(\overline{x})|_p \neq 0$ while $R_1^t(P') = 0$, a contradiction.Therefore, we can throw away the portion of P' after time t (including the t-th instruction) and generate R_1's contents $R_1^t(P')$ by an offsetting matrix m' at time t, as follows:*

  We have that $R_1^t(P')$ is some non-zero multiple of $f(\overline{x})|_p$, say $R_1^t(P') = s \cdot f(\overline{x})|_p$. We also have that $R_1^t(P') = a \cdot R_1^{t-1}(P') + b \cdot R_2^{t-1}(P')$. If a is a unit, then the desired output $f(\overline{x})|_p$ is produced by the assignment $R_1^t(P') \leftarrow (a/s) \cdot R_1^{t-1}(P') + (b/s) \cdot R_2^{t-1}(P')$, which can be accomplished by a rule of type 3(a) or 4(a) (since we do not care what value is placed in R_2). If b is a unit, then the desired assignment instead is produced by a transposition of a rule of type 3(b) or 4(b). Thus, in either case, we obtain a $\mu_{\mathbb{H}_{2\times 2}}$-SLP of size at most $t + 1$ for $f(\overline{x})|_p$.

Definition 85 *Let $f(\overline{x})$ be a polynomial and P be one of its $\mu_{\mathbb{H}_{2\times 2}}$-SLPs. We classify the potentially degenerate matrices m_P^t in P, if they do exist, according to the following criterion: If m_P^t has at least one finishing projection, then m_P^t is* good*; Otherwise, m_P^t is* bad*.*

In the same spirit, we can classify inherently degenerate matrices in $\mu_{\mathbb{H}_{2\times 2}}$-SLPs. In this case, we can consider the degenerating projection to be the empty set. The following lemma is essentially a variant of Lemma 81. Hence, we omit its proof.

Lemma 86 *Let $f(\overline{x})$ be a nonzero polynomial under any well-formed regular projection of size at most four and P be one of its $\mu_{\mathbb{H}_{2\times 2}}$-SLPs. Suppose that there exists $0 < t \leq |P|$ such that m_P^t is an inherently degenerate matrix. Let $w \in \mathbb{F}$. Then up to the permutation of the indices, only one of the two following cases will happen.*

1. *$R_1^t \in \mathbb{F}^*$ and $R_2^t = w \cdot R_1^t$ for some $w \in \mathbb{F}$.*
2. *R_1^t has degree at least one, $R_1^t = w \cdot R_2^t$ and $f(\overline{x})$ is divisible by R_1^t, where $w \in \mathbb{F}$.*

Furthermore, If $f(\overline{x})$ is a nonzero irreducible polynomial, then in the second case, $R_1^t = c \cdot f(\overline{x})$ where $c \in \mathbb{F}^*$.

Definition 87 *Let $f(\overline{x})$ be as in Lemma 86 and let P be one of its $\mu_{\mathbb{H}_{2\times 2}}$-SLPs. Let m_t be an inherently degenerate matrix in P if it exists. If the first case in Lemma 86 happens, then we say that m_t is* bad*, otherwise, it is* good*.*

Note that the notions of badness and goodness apply only to potentially and inherently degenerate matrices.

Observation 88 *Let p be an arbitrary homogeneous projection and let $P' = P|_p$. If m_P^t is bad in P, then $m_{P'}^t$ can not be good in P' (This is because, if m_P^t is bad, then under any extension of p, at time t both registers compute field elements which are constant polynomials with no variables). More precisely, $m_{P'}^t$ either stays as a bad matrix or becomes an inherently non-degenerate matrix. Furthermore, inherently non-degenerate matrices will never be turned into some other type by any projection.*

Now we are ready to present our main impossibility theorem of this section.

Theorem 89 *If $k \geq 8$, then $f(\overline{x}) = \sum_{i=1}^{k} x_{2i-1}x_{2i}$ is not computable by $\mathbb{H}_{2\times 2}$. That is, for every n, $f(\overline{x})$ can not be obtained from $IMM_{2,n}$ under homogeneous projections.*

Proof. We prove the theorem by contradiction. Suppose P is a $\mu_{\mathbb{H}_{2\times 2}}$-SLP for $f(\overline{x})$. We define the set G of time steps as:

$$G = \{t \mid m_P^t \text{ is a good matrix}\}.$$

There are two cases to consider.

- The first case is that $G = \emptyset$. Define the set B similarly as:

$$B = \{t \mid m_P^t \text{ is a bad matrix}\}.$$

If B is empty as well, then P is indeed a $\mu_{\mathbb{H}_{2\times 2}\cap\text{Indg}}$-SLP. By Fact 60, the highest-degree homogeneous part of $f(\overline{x})$ is irreducible, and by Theorem 74, we have reached a contradiction. Otherwise, let $t_B = \max(B)$. Note that, by Proposition 65, the output at time $|P|$ is a non-zero polynomial under any well-formed regular projection of size at most six, which means that $m_P^{|P|}$ cannot be bad, and hence $t_B < |P|$. Let p be one of the cutting projections of $m_P^{t_B}$. Consider $P|_p$ and the polynomial $f(\overline{x})|_p$ it computes. Since the size of p is bounded by six, by Proposition 65, $f(\overline{x})|_p$ is again an irreducible polynomial and moreover, its degree-two homogeneous part is irreducible. For all t such that $t_B \leq t \leq |P|$, m_P^t is an inherently non-degenerate matrix. By the first item of Observation 84, we now have a $\mu_{\mathbb{H}_{2\times 2}\cap\text{Indg}}$-SLP for $f(\overline{x})|_p$ which is a contradiction to Theorem 74. Notice that by Proposition 65, the above arguments apply to any polynomial of the form $\sum_{i=1}^{5} x_{2i-1}x_{2i} + l(\overline{x})$ where $l(\overline{x})$ is an arbitrary linear function.

- We assume that $G \neq \emptyset$. Let $t_G = \min G$. Suppose first that $m_P^{t_G}$ is an inherently degenerate matrix. Since $m_P^{t_G}$ is good, we have by Lemma 86 that, at time t_G, register R_1 computes a nonzero multiple of f. Hence by the second item of Observation 84 with $p = \emptyset$, we obtain a new $\mu_{\mathbb{H}_{2\times 2}}$-SLP for $f(\overline{x})$, consisting of only the matrices before t_G – none of which are good. This brings us back to the first case and a contradiction.

  Otherwise, assume that $m_P^{t_G}$ is a potentially degenerate matrix and let p be one of its finishing projections of size at most six. Consider $P|_p$ and the polynomial $f(\overline{x})|_p$ it computes. By the second item of Observation 84, we obtain a new $\mu_{\mathbb{H}_{2\times 2}}$-SLP P' for $f(\overline{x})|_p$ and furthermore, by Observation 88, P' does not contain any good matrices. Hence, this reduces us to the first case, since $f(\overline{x})|_p$ is of the form $\sum_{i=1}^{5} x_{2i-1}x_{2i} + l(\overline{x})$ where $l(\overline{x})$ is an arbitrary linear function. It is not hard to see that we will arrive at a contradiction for $f(\overline{x})|_p$, which completes our proof.

□

The proof of Theorem 89 leads to the following corollary.

Corollary 90 *If $k \geq 8$, then $f(\overline{x}) = \sum_{i=1}^{k} x_{2i-1}x_{2i} + l(\overline{x})$ is not computable by $\mathbb{H}_{2\times 2}$, where $l(\overline{x})$ is an arbitrary linear function.*

4.3 Extensions to simple and regular projections

In this section, we show that in the seemingly more powerful models, it is still hard to compute simple polynomials. We start by extending the result of Section 4.2 to the case of simple projections. Then by similar techniques and some extra observations, we will prove that certain polynomials are not regular projections of $\text{IMM}_{2,n}$, and thus, they are not computable by algebraic branching programs of width two.

4.3.1 Impossibility result for simple projections

In order to show that an analogue of Theorem 74 holds in the setting of simple projections, we first show that, for nondegenerate matrices, the simple case reduces to the homogeneous case.

Lemma 91 *Every matrix in $\mathbb{S}_{2\times 2} \cap$ Indg can be represented by a product of matrices in $\mathbb{H}_{2\times 2} \cap$ Indg.*

Proof. Let m be a matrix in $\mathbb{S}_{2\times 2}$ and

$$m = \begin{bmatrix} c_{1,1}y_{1,1} + w_{1,1} & c_{1,2}y_{1,2} + w_{1,2} \\ c_{2,1}y_{2,1} + w_{2,1} & c_{2,2}y_{2,2} + w_{2,2} \end{bmatrix},$$

where $c_{i,j}, w_{i,j} \in \mathbb{F}$ and $y_{i,j} \in \{x_k \mid k \in \mathbb{N}\}$.

We say that the variable $x_k = y_{i,j}$ *occurs in* m if $c_{i,j} \neq 0$ and that $y_{i,j}$ is an *occurrence for* x_k. Assume that $m \in \mathbb{S}_{2\times 2} \cap$ Indg and consider the following cases.

1. If there are no occurrences of any variables, then m is a linear transformation over $\mathbb{F}$. So $m \in \mathbb{H}_{2\times 2} \cap$ Indg.

2. If there are at least three distinct variables occurring in m, then $\det(m)$ is a nonzero polynomial and $m \in$ Pdg, a contradiction to our assumption.

3. If there is only a single variable x_k occurring in m, then obviously x_k has either two or four occurrences in m which can be divided into two subcases.

- If x_k has two occurrences in m, then these two occurrences can not be placed at the diagonal or anti-diagonal positions. Hence, without loss of generality, assume that m has the following form.

$$m = \begin{bmatrix} w_{1,1} & w_{1,2} \\ c_{2,1}x_k + w_{2,1} & c_{2,2}x_k + w_{2,2} \end{bmatrix},$$

where $c_{2,1} \neq 0$ and $c_{2,2} \neq 0$.

The determinant of m is equal to $(c_{2,2}w_{1,1} - c_{2,1}w_{1,2})x_k + (w_{1,1}w_{2,2} - w_{1,2}w_{2,1})$, then by our assumption, $c_{2,2}w_{1,1} - c_{2,1}w_{1,2} = 0$. If $w_{1,1} = w_{1,2} = 0$, then $m \in$ Idg, a contradiction to our assumption. If exactly one of them is equal to zero, then $m \in$ Pdg, a contradiction as well. Hence, we can assume that $\frac{c_{2,2}}{c_{2,1}} = \frac{w_{1,2}}{w_{1,1}} = d \neq 0$. Then,

$$m = \begin{bmatrix} w_{1,1} & 0 \\ c_{2,1}x_k + w_{2,1} & w_{2,2} - dw_{2,1} \end{bmatrix} \begin{bmatrix} 1 & d \\ 0 & 1 \end{bmatrix}$$

$w_{2,2} - dw_{2,1} \neq 0$ since $\det(m) \neq 0$.

So

$$m = \begin{bmatrix} \frac{w_{1,1}}{c_{2,1}} & 0 \\ x_k & w_{2,2} - dw_{2,1} \end{bmatrix} \begin{bmatrix} c_{2,1} & 0 \\ \frac{w_{2,1}}{w_{2,2}-dw_{2,1}} & 1 \end{bmatrix} \begin{bmatrix} 1 & d \\ 0 & 1 \end{bmatrix}$$

This verifies that m is a product of matrices in $\mathbb{H}_{2\times 2} \cap$ Indg.

- If x_k has four occurrences in m, then assume m has the following form.

$$m = \begin{bmatrix} c_{1,1}x_k + w_{1,1} & c_{1,2}x_k + w_{1,2} \\ c_{2,1}x_k + w_{2,1} & c_{2,2}x_k + w_{2,2} \end{bmatrix},$$

where each $c_{i,j} \neq 0$.

The determinant of m is equal to $(c_{1,1}x_k + w_{1,1})(c_{2,2}x_k + w_{2,2}) - (c_{1,2}x_k + w_{1,2})(c_{2,1}x_k + w_{2,1})$. Because $m \in$ Indg, $c_{1,1}c_{2,2} - c_{1,2}c_{2,1} = 0$. Let $d = \frac{c_{1,2}}{c_{1,1}} = \frac{c_{2,2}}{c_{2,1}} \neq 0$. Then, there exists $u, v \in \mathbb{F}$ such that

$$m = \begin{bmatrix} c_{1,1}x_k + w_{1,1} & u \\ c_{2,1}x_k + w_{2,1} & v \end{bmatrix} \begin{bmatrix} 1 & d \\ 0 & 1 \end{bmatrix}$$

Obviously the first matrix belongs to Indg because its determinant belongs to $\mathbb{F}^*$. Hence by the first subcase, it is a product of matrices in $\mathbb{H}_{2\times 2} \cap$ Indg, so is m.

4. If there are exactly two distinct variables x_k and x_l occurring in m, then they must have the same number of occurrences in m. Let $c_{i,j} \in \mathbb{F}^*$. It is clear that for all $u, v \in \mathbb{F}$, up to the permutation of rows and columns, the following matrices can not belong to Indg.

$$\begin{bmatrix} c_{1,1}x_k + w_{1,1} & u \\ c_{2,1}x_j + w_{2,1} & v \end{bmatrix}, \begin{bmatrix} c_{1,1}x_k + w_{1,1} & u \\ v & c_{2,1}x_j + w_{2,1} \end{bmatrix}$$

Hence, each of x_k and x_l has two occurrences. Without loss of generality, m has the following form.

$$m = \begin{bmatrix} c_{1,1}x_k + w_{1,1} & c_{1,2}x_k + w_{1,2} \\ c_{2,1}x_j + w_{2,1} & c_{2,2}x_j + w_{2,2} \end{bmatrix}.$$

Since $\det(m) \in \mathbb{F}^*$, we have $c_{1,1}c_{2,2} - c_{1,2}c_{2,1} = 0$. Let $d = \frac{c_{1,2}}{c_{1,1}} = \frac{c_{2,2}}{c_{2,1}} \neq 0$. Then, there exists $u, v \in \mathbb{F}$ such that

$$m = \begin{bmatrix} c_{1,1}x_k + w_{1,1} & u \\ c_{2,1}x_j + w_{2,1} & v \end{bmatrix} \begin{bmatrix} 1 & d \\ 0 & 1 \end{bmatrix}$$

This implies that m can not be an inherently non-degenerate matrix, a contradiction.

In conclusion, we have proven our claim that every matrix in $\mathbb{S}_{2\times 2} \cap$ Indg is equal to a product of matrices in $\mathbb{H}_{2\times 2} \cap$ Indg. □

The preceding lemma, together with Theorem 74, immediately yield the following corollary:

Corollary 92 *Let $f(\overline{x})$ be a polynomial whose highest-degree homogeneous part is irreducible. Then $f(\overline{x})$ is not computable by $\mathbb{S}_{2\times 2} \cap$ Indg.*

Next we show how to adapt the machinery in Section 4.2.3 and prove a similar impossibility theorem in terms of simple projections.

Theorem 93 *Let $l(\overline{x})$ be an arbitrary linear function. If $k \geq 8$, then $f(\overline{x}) = \sum_{i=1}^{k} x_{2i-1}x_{2i} + l(\overline{x})$ is not computable by $\mathbb{S}_{2\times 2}$, namely, for any n, $f(\overline{x})$ can not be obtained from $IMM_{2,n}$ under simple projections.*

Proof. [Proof sketch] We prove the theorem via contradiction. Suppose there is a $\mu_{\mathbb{S}_{2\times 2}}$-SLP P for $f(\overline{x})$.

Similar to Lemma 78, we prove the following lemma.

Lemma 94 *There does not exist a matrix m in P such that each entry of m contains a distinct variable. This implies that all matrices in P must contain at most three variables.*

Proof. Suppose the statement is not true, and without loss of generality,

$$m = \begin{bmatrix} c_1x_1 - w_1 & c_2x_2 - w_2 \\ c_3x_3 - w_3 & c_4x_4 - w_4 \end{bmatrix},$$

where $\forall 1 \leq i \leq 4, c_i \in \mathbb{F}^*, w_i \in \mathbb{F}$ and the x_is are all distinct.

Consider the projection $p = \{x_i \leftarrow \frac{w_i}{c_i} \mid 1 \leq i \leq 4\}$. Then $f(\overline{x})|_p$ is nonzero while $P|_p$ contains $\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$. By Propositions 64 and 65, $f(\overline{x})$ is nonzero under any regular projection of size at most four. Thus by Lemma 77, we have reached a contradiction. □

A direct consequence of Lemma 94 is an analogue of Lemma 79. The proof of the following lemma proceeds in the same way as that of Lemma 79, so we omit it here.

Lemma 95 *For any matrix $m \in P$ which belongs to $\mathbb{S}_{2\times 2} \cap$ Pdg, there exists a homogeneous projection p of size at most three such that $m|_p$ is degenerate and all of the entries in $m|_p$ belong to $\mathbb{F}$. Moreover, there is a well-formed homogeneous projection q of size at most six extending p.*

The notions of degenerating projections, and of good and bad matrices, thus carry over also to the setting of simple projections, and the rest of the proof follows exactly as in Section 4.2.3.

□

4.3.2 Impossibility result for regular projections

Let $m \in \mathbb{R}_{2\times 2}$ be of the following form:

$$m = \begin{bmatrix} l_{1,1} + w_{1,1} & l_{1,2} + w_{1,2} \\ l_{2,1} + w_{2,1} & l_{2,2} + w_{2,2} \end{bmatrix}.$$

where $w_{i,j} \in \mathbb{F}$ and the $l_{i,j}$'s are homogeneous linear forms in $\{\sum_{k=1}^{n} c_k x_k \mid n \in \mathbb{N}, c_k \in \mathbb{F}\}$. We will pay attention to the rank of the subspace spanned by $\{l_{i,j} \mid i,j \in \{1,2\}\}$, denoted as $r(m)$, which in some sense characterizes the number of "independent variables" among the $l_{i,j}$'s.

The following lemma illustrates the sense in which we can treat linearly-independent homogeneous linear forms as independent variables.

Lemma 96 *Let $l_1, l_2, \ldots, l_k$ be k linearly independent homogeneous linear forms, and let $w_1, \ldots, w_k$ be elements of $\mathbb{F}$. Then there is a regular projection p of size k such that, for all i, $l_i|_p = w_i$. (Thus we can think of p as a "projection" of the form $\{l_i \leftarrow w_i\}$.)*

Proof. The homogeneous linear form l_1 is of the form $\sum_{j=1}^{n} c_j x_j$, where each $c_j \in \mathbb{F}^*$. Start building the projection p with the rule $x_1 \leftarrow (w_1 - \sum_{j=2}^{n} c_j x_j)/c_1$. This clearly has the effect that $l_1|_p = w_1$. If $k = 1$, then the construction ends here.

Otherwise, let $l_2 = \sum_{j=1}^{n'} d_j y_j$, where each $d_j \in \mathbb{F}^*\}$. If the variable x_1 appears as one of the variables y_j, then replace x_1 with the expression $(w_1 - \sum_{j=2}^{n} c_j x_j)/c_1$ and simplify. By linear independence, there must still be some variable remaining in the resulting expression. Without loss of generality, let the resulting expression be of the form $\sum_{j=2}^{n''} a_j x_j$. Then we add a new rule $x_2 \leftarrow (w_2 - \sum_{j=3}^{n''} a_j x_j)/a_2$ (and if this variable x_2 occurs in the right-hand-side of the rule for x_1, then substitute this expression in for x_2 in that rule, and simplify). At this point, we have $l_1|_p = w_1$ and $l_2|_p = w_2$.

We continue in this way for all of the remaining linear forms. The crucial observation is that there will always be a variable in each linear form $l_j|p$ when we first consider it, because of linear independence. □

Our next lemma is a generalization of Lemma 91.

Lemma 97 *Every matrix m in $\mathbb{R}_{2\times 2} \cap$ Indg can be represented by a product of matrices in $\mathbb{H}_{2\times 2} \cap$ Indg.*

Proof. The correctness of the following claim is easy to verify.

If $r(m) = 0, 3$ or 4, then the proof is completely analogous to the corresponding cases in

Lemma 91, where we do our case analysis based on $r(m)$ instead of the number of variables that occur in m.

If $r(m) = 1$, then there exists a homogeneous linear form l such that all $l_{i,j}$s in m are multiples of l. By treating l as a single variable, the analysis of the third case in Lemma 91 reveals that m is a product of matrices from $\mathbb{H}_{2\times 2} \cap$ Indg as well as matrices having the following form:

$$\begin{bmatrix} c & 0 \\ l & c' \end{bmatrix}, \begin{bmatrix} c & l \\ 0 & c' \end{bmatrix}.$$

Thus the case when $r(m) = 1$ is completed by appealing to the following claim:

Claim 98 *Any matrix having the following form can be expressed as the product of matrices in* $\mathbb{H}_{2\times 2} \cap$ *Indg.*

$$\begin{bmatrix} c & 0 \\ l & c' \end{bmatrix}, \begin{bmatrix} c & l \\ 0 & c' \end{bmatrix}$$

where $c, c' \in \mathbb{F}^*$ *and* $l \in \mathbb{L}$.

Proof. We prove the claim by induction on the number of variables appearing in l. If l contains at most one variable, then the claim follows from Lemma 91.

Otherwise, l is of the form $dx_1 + l'$. Observe that

$$\begin{bmatrix} c & 0 \\ l & c' \end{bmatrix} = \begin{bmatrix} c/d & 0 \\ x_1 & 1 \end{bmatrix} \times \begin{bmatrix} d & 0 \\ l' & c' \end{bmatrix}.$$

(The other case is similar.) The claim now follows by induction. □

If $r(m) = 2$, then let $l_1, l_2 \in \{l_{i,j} \mid i, j \in \{1, 2\}\}$ be a basis. If every $l_{i,j}$ is a multiple of either l_1 or l_2, then the proof of the fourth case of Lemma 91 provides us a contradiction. Thus, we only need to consider the case where there is at least one $l' \in \{l_{i,j} \mid i, j \in \{1, 2\}\}$ and $c, c' \in \mathbb{F}^*$ such that $l' = cl_1 + c'l_2$, which means that l' is a non-trivial linear combination of l_1 and l_2. Therefore, without loss of generality, we assume that m has the following form.

$$m = \begin{bmatrix} l_1 + w_{1,1} & l_2 + w_{1,2} \\ cl_1 + c'l_2 + w_{2,1} & dl_1 + d'l_2 + w_{2,2} \end{bmatrix}.$$

where $c, c' \in \mathbb{F}^*, d, d' \in \mathbb{F}$.

But then the degree-two homogeneous part of $\det(m)$ is equal to $dl_1^2 + (d' - c)l_1 l_2 - c' l_2^2$, which is nonzero since $c' \neq 0$. This contradicts our assumption that $m \in$ Indg. □

The preceding lemma, together with Theorem 74, immediately yield the following corollary:

Corollary 99 *Let $f(\overline{x})$ be a polynomial whose highest-degree homogeneous part is irreducible. Then $f(\overline{x})$ is not computable by $\mathbb{R}_{2\times 2} \cap$ Indg.*

Now we are ready to prove our main theorem.

Theorem 100 (Theorem 11 restated) *If $k \geq 8$, then $f(\overline{x}) = \sum_{i=1}^{k} x_{2i-1}x_{2i}$ is not computable by $\mathbb{R}_{2\times 2}$, namely, for any n, $f(\overline{x})$ can not be obtained from IMM$_{2,n}$ under regular projections.*

Proof. [Proof sketch] The proof is by contradiction. Suppose P is a $\mu_{\mathbb{R}_{2\times 2}}$-SLP for $f(\overline{x})$.

By Propositions 64 and 65, $f(\overline{x})$ is nonzero under any regular projection of size at most four, which leads to the following lemma.

Lemma 101 *For any potentially degenerate matrix m_P^t in P, $r(m_P^t) \leq 3$.*

Proof. Suppose $r(m_P^t) = 4$, then there exists a regular projection p of size four such that $\forall 1 \leq i, j \leq 2,\ l_{i,j} = -w_{i,j}$. In other words, $m_P^t|_p = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$. By Lemma 77, we have reached a contradiction to the aforementioned property of $f(\overline{x})$. □

Lemma 102 *Any potentially degenerate matrix m_P^t in P has a regular projection p of size at most three such that $m_P^t|_p$ is degenerate and all of the entries in $m|_p$ belong to $\mathbb{F}$. Moreover, there is a well-formed regular projection q of size at most six extending p.*

Proof. $\det(m_P^t)$ is a polynomial of degree at least one. by Lemma 101, after some suitable linear transformation, $\det(m_P^t)$ can be viewed as a polynomial in at most three "new" variables that are linear forms in terms of the original set of variables. Since $\mathbb{F}$ is algebraically closed, this provides us the desired projection p. The second part of the claim follows from Proposition 64. □

By Lemma 102, we can define degenerating projections in terms of well-formed regular projections. Note that the proofs of Lemma 81 and Corollary 82 hold, regardless of the type of projections. Thus we can also extend the definitions of cutting and finishing projections to well-formed regular projections. The following observation is a slight variant of Observation 84.

Observation 103 *Let $f(\overline{x})$ be a polynomial such that under any well-formed regular projection q of size at most six, $f(\overline{x})|_q$ is always a nonzero irreducible polynomial. Let P be a $\mu_{\mathbb{R}_{2\times 2}}$-SLP for $f(\overline{x})$ and let m_P^t be a potentially degenerate matrix in P. Let p be one of degenerating regular projections of m_P^t and let $P' = P|_p$. Then, a $\mu_{\mathbb{R}_{2\times 2}}$-SLP can be constructed for $f(\overline{x})|_p$.*

- *If p is a cutting projection for m_P^t in P, this case is identical to the first case in Observation 84.*
- *If p is a finishing projection for m_P^t in P, this case is identical to the second case in Observation 84.*

Now the remaining part of proof proceeds exactly as in Section 4.2.3, since it does not depend on the type of underlying projections at all, namely, regardless of whether they are homogeneous, simple or regular. □

References

[AAB+99] A. Ambainis, E. Allender, D. A. M. Barrington, S. Datta, and H. LêThanh. Bounded depth arithmetic circuits: Counting and closure. In *Proc. of International Conference on Automata, Languages, and Programming (ICALP)*, number 1644 in Lecture Notes in Computer Science, pages 149–158. Springer, 1999.

[AAD00] Manindra Agrawal, Eric Allender, and Samir Datta. On TC^0, AC^0, and arithmetic circuits. *Journal of Computer and System Sciences*, 60(2):395–421, 2000.

[AB09] Sanjeev Arora and Boaz Barak. *Computational Complexity, a modern approach*. Cambridge University Press, 2009.

[ABFR94] James Aspnes, Richard Beigel, Merrick L. Furst, and Steven Rudich. The expressive power of voting polynomials. *Combinatorica*, 14(2):135–148, 1994.

[ABKPM09] Eric Allender, Peter Bürgisser, Johan Kjeldgaard-Pedersen, and Peter Bro Miltersen. On the complexity of numerical analysis. *SIAM Journal on Computing*, 38(5):1987–2006, 2009.

[ACR98] Alexander E. Andreev, Andrea E. F. Clementi, and José D. P. Rolim. A new general derandomization method. *Journal of the ACM*, 45(1):179–213, 1998.

[ACR99] Alexander E. Andreev, Andrea E. F. Clementi, and José D. P. Rolim. Worst-case hardness suffices for derandomization: A new method for hardness-randomness trade-offs. *Theoretical Computer Science*, 221(1-2):3–18, 1999.

[ACRT99] Alexander E. Andreev, Andrea E. F. Clementi, José D. P. Rolim, and Luca Trevisan. Weak random sources, hitting sets, and BPP simulations. *SIAM Journal on Computing*, 28(6):2103–2116, 1999.

[Adl78] Leonard M. Adleman. Two theorems on random polynomial time. In *Proc. IEEE Symp. on Found. of Comp. Sci. (FOCS)*, pages 75–83, 1978.

[ADR05] Eric Allender, Samir Datta, and Sambuddha Roy. Topology inside NC^{1^1}. In *Proc. IEEE Conf. on Computational Complexity*, pages 298–307, 2005.

[AG94] Eric Allender and Vivek Gore. A uniform circuit lower bound for the permanent. *SIAM Journal on Computing*, 23(5):1026–1049, 1994.

[AH94] Eric Allender and Ulrich Hertrampf. Depth reduction for circuits of unbounded fan-in. *Information and Computation*, 112(2):217–238, 1994.

[AHM+08] Eric Allender, Lisa Hellerstein, Paul McCabe, Toniann Pitassi, and Michael E. Saks. Minimizing disjunctive normal form formulas and AC^0 circuits given a truth table. *SIAM Journal on Computing*, 38(1):63–84, 2008.

[AJMV98] Eric Allender, Jia Jiao, Meena Mahajan, and V. Vinay. Non-commutative arithmetic circuits: Depth reduction and size lower bounds. *Theor. Comput. Sci.*, 209(1-2):47–86, 1998.

[Ajt83] Miklós Ajtai. σ_1^1-formulae on finite structures. *Annals of Pure and Applied Logic*, 24:1–48, 1983.

[AK97] Vikraman Arvind and Johannes Köbler. On resource-bounded measure and pseudorandomness. *Proc. Conference on Foundations of Software Technology and Theoretical Computer Science (FST&TCS)*, pages 235–249, 1997.

[AK10] Eric Allender and Michal Koucký. Amplifying lower bounds by means of self-reducibility. *Journal of the ACM*, 57(3), 2010.

[All89] Eric Allender. A note on the power of threshold circuits. In *Proc. IEEE Symp. on Found. of Comp. Sci. (FOCS)*, pages 580–584, 1989.

[All96] Eric Allender. Circuit complexity before the dawn of the new millennium. In *Proc. Conference on Foundations of Software Technology and Theoretical Computer Science (FST&TCS)*, pages 1–18, 1996.

[All99] Eric Allender. The permanent requires large uniform threshold circuits. *Chicago Journal of Theorerical Computer Science*, 1999, 1999.

[All04] Eric Allender. Arithmetic circuits and counting complexity classes. In J. Krajíček, editor, *Complexity of Computations and Proofs*, volume 13 of *Quaderni di Matematica*, pages 33–72. Seconda Università di Napoli, 2004.

[All10] Eric Allender. New surprises from self-reducibility. In F. Ferreira, H. Guerra, E. Mayordomo, and J. Rasga, editors, *Programs, Proofs, Processes*, Conference on Computability in Europe, pages 1–5. Centre for Applied Mathematics and Information Technology, Dept. of Mathematics, University of Azores, 2010.

[AV08] Manindra Agrawal and V. Vinay. Arithmetic circuits: A chasm at depth four. In *Proc. IEEE Symp. on Found. of Comp. Sci. (FOCS)*, pages 67–75, 2008.

[AW93] Eric W. Allender and Klaus W. Wagner. Counting hierarchies: polynomial time and constant depth circuits. In G. Rozenberg and A. Salomaa, editors, *Current Trends in Theoretical Computer Science*, World Scientific Series in Computer Science, pages 469–483. World Scientific Press, 1993.

[Bar85] David A. Barrington. Width-3 permutation branching programs. Technical Report Technical Memorandum MIT/LCS/TM-293, MIT, 1985.

[Bar89] David A. Barrington. Bounded-width polynomial-size branching programs recognize exactly those languages in NC^1. *Journal of Computer and System Sciences*, 38(1):150–164, 1989.

[Bar92] David A. Mix Barrington. Quasipolynomial size circuit classes. In *Proc. IEEE Conf. on Structure in Complexity Theory*, pages 86–93, 1992.

[Bea] Paul Beame. A switching lemma primer. Manuscript http://www.cs.washington.edu/homes/beame/primer.ps.

[Bei94] Richard Beigel. When do extra majority gates help? polylog(n) majority gates are equivalent to one. *Computational Complexity*, 4:314–324, 1994.

[BF99] Harry Buhrman and Lance Fortnow. One-sided versus two-sided error in probabilistic computation. *Proc. of Symp. on Theo. Aspects of Comp. Sci. (STACS)*, pages 100–109, 1999.

[BFNW93] 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.

[BH89] Paul Beame and Johan Håstad. Optimal bounds for decision problems on the crcw pram. *Journal of the ACM*, 36(3):643–670, 1989.

[BIS90] David A. Mix Barrington, Neil Immerman, and Howard Straubing. On uniformity within NC^1. *Journal of Computer and System Sciences*, 41(3):274–306, 1990.

[BM84] Manuel Blum and Silvio Micali. How to generate cryptographically strong sequences of pseudo-random bits. *SIAM Journal on Computing*, 13(4):850–864, 1984.

[BOC88] Michael Ben-Or and Richard Cleve. Computing algebraic formulas using a constant number of registers. In *Proc. ACM Symp. on Theory of Computing (STOC)*, pages 254–257, 1988.

[BOC92] Michael Ben-Or and Richard Cleve. Computing algebraic formulas using a constant number of registers. *SIAM Journal on Computing*, 21(1):54–58, 1992.

[BT88] David A. Mix Barrington and Denis Thérien. Finite monoids and the fine structure of nc^1. *Journal of the ACM*, 35(4):941–952, 1988.

[BT94] Richard Beigel and Jun Tarui. On ACC. *Computational Complexity*, 4:350–366, 1994.

[BTT92] Richard Beigel, Jun Tarui, and Seinosuke Toda. On probabilistic acc circuits with an exact-threshold output gate. In *International Symposium on Algorithms and Computation*, pages 420–429, 1992.

[Bür00] Peter Bürgisser. Cook's versus valiant's hypothesis. *Theoretical Computer Science*, 235(1):71–88, 2000.

[Bus93] Samuel R. Buss. Algorithms for boolean formula evaluation and for tree-contraction. In P. Clote and J. Krajíček, editors, *Proof Theory, Complexity, and Arithmetic*, pages 95–115. Oxford University Press, 1993.

[Cai89] Jinyi Cai. With probability one, a random oracle separates PSPACE from the polynomial-time hierarchy. *Journal of Computer and System Sciences*, 38(1):68–85, 1989.

[CMTV98] H. Caussinus, P. McKenzie, D. Thérien, and H. Vollmer. Nondeterministic NC^0 computation. *Journal of Computer and System Sciences*, 57:200–212, 1998.

[CSV84] Ashok K. Chandra, Larry J. Stockmeyer, and Uzi Vishkin. Constant depth reducibility. *SIAM Journal on Computing*, 13(2):423–439, 1984.

[DS05] Zeev Dvir and Amir Shpilka. Locally decodable codes with 2 queries and polynomial identity testing for depth 3 circuits. In *Proc. ACM Symp. on Theory of Computing (STOC)*, pages 592–601, 2005.

[DSY09] Zeev Dvir, Amir Shpilka, and Amir Yehudayoff. Hardness-randomness tradeoffs for bounded depth circuits. *SIAM Journal on Computing*, 39(4):1279–1293, 2009.

[FF91] Joan Feigenbaum and Lance Fortnow. On the random-self-reducibility of complete sets. In *Structure in Complexity Theory Conference*, pages 124–132, 1991.

[FSS84] Merrick L. Furst, James B. Saxe, and Michael Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical Systems Theory*, 17(1):13–27, 1984.

[GGH+07] Shafi Goldwasser, Dan Gutfreund, Alexander Healy, Tali Kaufman, and Guy N. Rothblum. Verifying and decoding in constant depth. In *Proc. ACM Symp. on Theory of Computing (STOC)*, pages 440–449, 2007.

[GK98] Dima Grigoriev and Marek Karpinski. An exponential lower bound for depth 3 arithmetic circuits. In *Proc. ACM Symp. on Theory of Computing (STOC)*, pages 577–582, 1998.

[GR00] Dima Grigoriev and Alexander A. Razborov. Exponential lower bounds for depth 3 arithmetic circuits in algebras of functions over finite fields. *Applicable Algebra in Engineering, Communication and Computing*, 10(6):465–487, 2000.

[Gro94] Vince Grolmusz. A weight-size trade-off for circuits with mod_m gates. In *Proc. ACM Symp. on Theory of Computing (STOC)*, pages 68–74, 1994.

[GV04] Dan Gutfreund and Emanuele Viola. Fooling parity tests with parity gates. *APPROX-RANDOM*, pages 381–392, 2004.

[GVW00] Oded Goldreich, Salil P. Vadhan, and Avi Wigderson. Simplified derandomization of BPP using a hitting set generator. *Electronic Colloquium on Computational Complexity*, 7(4), 2000.

[GW99] Oded Goldreich and Avi Wigderson. Improved derandomization of BPP using a hitting set generator. *RANDOM-APPROX*, pages 131–137, 1999.

[HAB02] William Hesse, Eric Allender, and David A. Mix Barrington. Uniform constant-depth threshold circuits for division and iterated multiplication. *Journal of Computer and System Sciences*, 65(4):695–716, 2002.

[Han06] Kristoffer Arnsfelt Hansen. Constant width planar computation characterizes ACC^0. *Theory of Computing Systems*, 39(1):79–92, 2006.

[Hås86] Johan Håstad. Almost optimal lower bounds for small depth circuits. In *Proc. ACM Symp. on Theory of Computing (STOC)*, pages 6–20, 1986.

[HG91] Johan Håstad and Mikael Goldmann. On the power of small-depth threshold circuits. *Computational Complexity*, 1:113–129, 1991.

[HK09] Kristoffer Arnsfelt Hansen and Michal Koucký. A new characterization of ACC^0 and probabilistic CC^0. In *Proc. IEEE Conf. on Computational Complexity*, pages 27–34, 2009.

[HM04] Kristoffer Arnsfelt Hansen and Peter Bro Miltersen. Some meet-in-the-middle circuit lower bounds. In *Proc. of Math. Foundations of Comp. Sci. (MFCS)*, pages 334–345, 2004.

[HMP+93] András Hajnal, Wolfgang Maass, Pavel Pudlák, Mario Szegedy, and György Turán. Threshold circuits of bounded depth. *Journal of Computer and System Sciences*, 46(2):129–154, 1993.

[IKW02] Russell Impagliazzo, Valentine Kabanets, and Avi Wigderson. In search of an easy witness: exponential time vs. probabilistic polynomial time. *Journal of Computer and System Sciences*, 65(4):672–694, 2002.

[Imm89] Neil Immerman. Expressibility and parallel complexity. *SIAM Journal on Computing*, 18(3):625–638, 1989.

[Imp95] Russell Impagliazzo. Hard-core distributions for somewhat hard problems. *Proc. IEEE Symp. on Found. of Comp. Sci. (FOCS)*, pages 538–545, 1995.

[IPS97] Russell Impagliazzo, Ramamohan Paturi, and Michael E. Saks. Size-depth tradeoffs for threshold circuits. *SIAM Journal on Computing*, 26(3):693–707, 1997.

[ISW06] Russell Impagliazzo, Ronen Shaltiel, and Avi Wigderson. Reducing the seed length in the Nisan-Wigderson generator. *Combinatorica*, 26(6):647–681, 2006.

[IW97] Russell Impagliazzo and Avi Wigderson. P = BPP if E requires exponential circuits: Derandomizing the XOR lemma. *Proc. ACM Symp. on Theory of Computing (STOC)*, pages 220–229, 1997.

[IW01] Russell Impagliazzo and Avi Wigderson. Randomness vs time: Derandomization under a uniform assumption. *Journal of Computer and System Sciences*, 63(4):672–688, 2001.

[Jan08] Maurice J. Jansen. Lower bounds for syntactically multilinear algebraic branching programs. In *Proc. of Math. Foundations of Comp. Sci. (MFCS)*, number 5162 in Lecture Notes in Computer Science, pages 407–418. Springer, 2008.

[JR09] Maurice J. Jansen and B. V. Raghavendra Rao. Simulation of arithmetical circuits by branching programs with preservation of constant width and syntactic multilinearity. In *CSR*, number 5675 in Lecture Notes in Computer Science, pages 179–190. Springer, 2009.

[Jun85] H. Jung. Depth efficient transformations of arithmetic into Boolean circuits. In *Proc. FCT*, number 199 in Lecture Notes in Computer Science, pages 167–173. Springer, 1985.

[Kab02] Valentine Kabanets. Derandomization: a brief overview. *Bulletin of the EATCS*, 76:88–103, 2002.

[KI04] Valentine Kabanets and Russell Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Computational Complexity*, 13(1-2):1–46, 2004.

[Kil88] Joe Kilian. Founding cryptography on oblivious transfer. In *Proc. ACM Symp. on Theory of Computing (STOC)*, pages 20–31, 1988.

[KL80] Richard M. Karp and Richard J. Lipton. Some connections between nonuniform and uniform complexity classes. In *Proc. ACM Symp. on Theory of Computing (STOC)*, pages 302–309, 1980.

[KMSV10] Zohar Shay Karnin, Partha Mukhopadhyay, Amir Shpilka, and Ilya Volkovich. Deterministic identity testing of depth-4 multilinear circuits with bounded top fan-in. In *Proc. ACM Symp. on Theory of Computing (STOC)*, pages 649–658, 2010.

[KP94] Matthias Krause and Pavel Pudlák. On the computational power of depth 2 circuits with threshold and modulo gates. In *Proc. ACM Symp. on Theory of Computing (STOC)*, pages 48–57, 1994.

[KP09] Pascal Koiran and Sylvain Perifel. A superpolynomial lower bound on the size of uniform non-constant-depth threshold circuits for the permanent. In *Proc. IEEE Conf. on Computational Complexity*, pages 35–40, 2009.

[KS01] Adam Klivans and Daniel A. Spielman. Randomness efficient identity testing of multivariate polynomials. In *Proc. ACM Symp. on Theory of Computing (STOC)*, pages 216–223, 2001.

[KS06] Neeraj Kayal and Nitin Saxena. Polynomial identity testing for depth 3 circuits. In *Proc. IEEE Conf. on Computational Complexity*, pages 9–17, 2006.

[KS08] Zohar Shay Karnin and Amir Shpilka. Black box polynomial identity testing of generalized depth-3 arithmetic circuits with bounded top fan-in. In *Proc. IEEE Conf. on Computational Complexity*, pages 280–291, 2008.

[KS09] Neeraj Kayal and Shubhangi Saraf. Blackbox polynomial identity testing for depth 3 circuits. In *Proc. IEEE Symp. on Found. of Comp. Sci. (FOCS)*, pages 198–207, 2009.

[KvM02] Adam Klivans and Dieter van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses. *SIAM Journal on Computing*, 31(5):1501–1526, 2002.

[KVVY93] Ravi Kannan, H. Venkateswaran, V. Vinay, and Andrew Chi-Chih Yao. A circuit-based proof of toda's theorem. *Information and Computation*, 104(2):271–276, 1993.

[Lip91] Richard J. Lipton. New directions in testing. In *Distributed Computing and Cryptography*, volume 2 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 191–202. AMS, 1991.

[LZ77] R. Lipton and Y. Zalcstein. Word problems solvable in logspace. *Journal of the ACM*, 24:522–526, 1977.

[MC87] Pierre McKenzie and Stephen A. Cook. The parallel complexity of Abelian permutation group problems. *SIAM Journal on Computing*, 16(5):880–909, 1987.

[MR09] Meena Mahajan and B. V. Raghavendra Rao. Small-space analogues of valiant's classes. In *FCT*, number 5699 in Lecture Notes in Computer Science, pages 250–261. Springer, 2009.

[MSS11] M. Mahajan, N. Saurabh, and K. Sreenivasaiah. Counting paths in planar width 2 branching programs. Manuscript, 2011.

[MV97] M. Mahajan and V. Vinay. Determinant: Combinatorics, algorithms, and complexity. *Chicago Journal of Theoretical Computer Science*, (5), 1997.

[MV05] Peter Bro Miltersen and N. V. Vinodchandran. Derandomizing Arthur-Merlin games using hitting sets. *Computational Complexity*, 14(3):256–279, 2005.

[Nis91] Noam Nisan. Lower bounds for non-commutative computation (extended abstract). In *Proc. ACM Symp. on Theory of Computing (STOC)*, pages 410–418, 1991.

[NR04] Moni Naor and Omer Reingold. Number-theoretic constructions of efficient pseudo-random functions. *Journal of the ACM*, 51(2):231–262, 2004.

[NW94] Noam Nisan and Avi Wigderson. Hardness vs randomness. *Journal of Computer and System Sciences*, 49(2):149–167, 1994.

[NW97] Noam Nisan and Avi Wigderson. Lower bounds on arithmetic circuits via partial derivatives. *Computational Complexity*, 6(3):217–234, 1997.

[PS88] Ian Parberry and Georg Schnitger. Parallel computation with threshold functions. *Journal of Computer and System Sciences*, 36(3):278–302, 1988.

[PS89] Ian Parberry and Georg Schnitger. Relating boltzmann machines to conventional models of computation. *Neural Networks*, 2(1):59–67, 1989.

[Raz87] Alexander Razborov. Lower bounds on the size of bounded-depth networks over a complete basis with logical addition. *Mathematical Notes of the Academy of Sciences. of the USSR*, 41(4):333–338, 1987.

[Raz92] Alexander A. Razborov. On small depth threshold circuits. In *Scandinavian Workshop on Algorithm Theory*, pages 42–52, 1992.

[Raz95] Alexander Razborov. Bounded arithmetic and lower bounds in boolean complexity. *Feasible Mathematics II*, pages 344–386, 1995.

[Raz09] Ran Raz. Multi-linear formulas for permanent and determinant are of super-polynomial size. *Journal of the ACM*, 56(2), 2009.

[Raz10] Ran Raz. Elusive functions and lower bounds for arithmetic circuits. *Theory of Computing*, 6(1):135–177, 2010.

[Rob93] D. Robinson. *Parallel algorithms for group word problems*. PhD thesis, Univ. of California, San Diego, 1993.

[RR97] Alexander A. Razborov and Steven Rudich. Natural proofs. *Journal of Computer and System Sciences*, 55(1):24–35, 1997.

[RW93] Alexander A. Razborov and Avi Wigderson. $n^{\Omega(logn)}$ lower bounds on the size of depth-3 threshold circuits with and gates at the bottom. *Information Processing Letters*, 45(6):303–307, 1993.

[RY09] Ran Raz and Amir Yehudayoff. Lower bounds and separations for constant depth multilinear circuits. *Computational Complexity*, 18(2):171–207, 2009.

[Sch80] Jacob T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *Journal of the ACM*, 27(4):701–717, 1980.

[Sha81] Adi Shamir. On the generation of cryptographically strong pseudo-random sequences. *Proc. of International Conference on Automata, Languages, and Programming (ICALP)*, pages 544–550, 1981.

[She07] Alexander A. Sherstov. Separating AC^0 from depth-2 majority circuits. In *Proc. ACM Symp. on Theory of Computing (STOC)*, pages 294–301, 2007.

[Shp01] Amir Shpilka. *Lower bounds for small depth arithmetic and Boolean circuits*. PhD thesis, Hebrew University, 2001.

[Sit96] Meera Sitharam. Approximation from linear spaces and applications to complexity. *Electronic Colloquium on Computational Complexity*, 3(30), 1996.

[Smo87] Roman Smolensky. Algebraic methods in the theory of lower bounds for boolean circuit complexity. In *Proc. ACM Symp. on Theory of Computing (STOC)*, pages 77–82, 1987.

[SS09] Nitin Saxena and C. Seshadhri. An almost optimal rank bound for depth-3 identities. In *Proc. IEEE Conf. on Computational Complexity*, pages 137–148, 2009.

[SS10] Nitin Saxena and C. Seshadhri. From sylvester-gallai configurations to rank bounds: Improved black-box identity test for depth-3 circuits. In *Proc. IEEE Symp. on Found. of Comp. Sci. (FOCS)*, pages 21–29, 2010.

[SS11] Nitin Saxena and C. Seshadhri. Blackbox identity testing for bounded top fanin depth-3 circuits: the field doesn't matter. In *Proc. ACM Symp. on Theory of Computing (STOC)*, 2011. To appear.

[STV01] Madhu Sudan, Luca Trevisan, and Salil P. Vadhan. Pseudorandom generators without the XOR lemma. *Journal of Computer and System Sciences*, 62(2):236–266, 2001.

[SU05] Ronen Shaltiel and Christopher Umans. Simple extractors for all min-entropies and a new pseudorandom generator. *Journal of the ACM*, 52(2):172–216, 2005.

[SV10] Ronen Shaltiel and Emanuele Viola. Hardness amplification proofs require majority. *SIAM Journal on Computing*, 39(7):3122–3154, 2010.

[SW01] Amir Shpilka and Avi Wigderson. Depth-3 arithmetic circuits over fields of characteristic zero. *Computational Complexity*, 10(1):1–27, 2001.

[Tar93] Jun Tarui. Probablistic polynomials, AC^0 functions, and the polynomial-time hierarchy. *Theoretical Computer Science*, 113(1):167–183, 1993.

[Tod89] Seinosuke Toda. On the computational power of pp and +p. In *Proc. IEEE Symp. on Found. of Comp. Sci. (FOCS)*, pages 514–519, 1989.

[Uma03] Christopher Umans. Pseudo-random generators for all hardnesses. *Journal of Computer and System Sciences*, 67(2):419–440, 2003.

[Val79a] L. Valiant. Completeness classes in algebra. In *Proc. ACM Symp. on Theory of Computing (STOC)*, pages 249–261, 1979.

[Val79b] Leslie G. Valiant. Completeness classes in algebra. In *Proc. ACM Symp. on Theory of Computing (STOC)*, pages 249–261, 1979.

[Vio05] Emanuele Viola. The complexity of constructing pseudorandom generators from hard functions. *Computational Complexity*, 13(3-4):147–188, 2005.

[vL99] Jacobus H. van Lint. *Introduction to Coding Complexity*. Springer-Verlag, 1999.

[Vol99] Heribert Vollmer. *Introduction to Circuit Complexity*. Springer, 1999.

[vzG87] Joachim von zur Gathen. Feasible arithmetic computations: Valiant's hypothesis. *Journal of Symbolic Computation*, 4(2):137–172, 1987.

[Wag86] Klaus W. Wagner. The complexity of combinatorial problems with succinct input representation. *Acta Informatica*, 23(3):325–356, 1986.

[Wil10] Ryan Williams. Improving exhaustive search implies superpolynomial lower bounds. In *Proc. ACM Symp. on Theory of Computing (STOC)*, pages 231–240, 2010.

[Wil11] Ryan Williams. Non-uniform ACC circuit lower bounds. In *Proc. IEEE Conf. on Computational Complexity*, 2011.

[Yao82] Andrew Chi-Chih Yao. Theory and applications of trapdoor functions (extended abstract). *Proc. IEEE Symp. on Found. of Comp. Sci. (FOCS)*, pages 80–91, 1982.

[Yao85] Andrew Chi-Chih Yao. Separating the polynomial-time hierarchy by oracles (preliminary version). In *Proc. IEEE Symp. on Found. of Comp. Sci. (FOCS)*, pages 1–10, 1985.

[Yao89] Andrew Chi-Chih Yao. Circuits and local computation. In *Proc. ACM Symp. on Theory of Computing (STOC)*, pages 186–196, 1989.

[Yao90] Andrew Chi-Chih Yao. On ACC and threshold circuits. In *Proc. IEEE Symp. on Found. of Comp. Sci. (FOCS)*, pages 619–627, 1990.

[Zak83] Stanislav Zak. A turing machine hierarchy. *Theoretical Computer Science*, 26:327–333, 1983.

[Zip79] Richard Zippel. Probabilistic algorithms for sparse polynomials. In *EUROSAM*, pages 216–226, 1979.

Vita

Fengming Wang

2003 B. Sc. in Computer Science, Nanjing University.

2006 M. Sc. in Computer Science, Iowa State University.

2011 M. Sc. in Mathematics, Rutgers University.

2011 Expected Ph.D. in Computer Science, Rutgers University.

2006 L. Fortnow, J. Hitchcock, A. Pavan, N. V. Vinodchandran, F. Wang. Extracting Kolmogorov Complexity with Applications to Dimension Zero-One Laws. In *Proc. of the 33rd International Conference on Automata, Languages, and Programming (ICALP)*, pp 335-345.

2007 A. Pavan, F. Wang. Robustness of PSPACE-complete sets. In *Information Processing Letters*, 103(3), pp 102-104.

2010 E. Allender, V. Arvind, F. Wang. Uniform Derandomization from Pathetic Lower Bounds. In *Proc. of 14th International Workshop on Randomization*, pp 380-393.

2011 L. Fortnow, J. Hitchcock, A. Pavan, N. V. Vinodchandran, F. Wang. Extracting Kolmogorov Complexity with Applications to Dimension Zero-One Laws. In *Information and Computation*, 209(4), pp 627-636.

2011 F. Wang. NEXP Does Not Have Non-uniform Quasipolynomial-Size ACC Circuits of $o(\log \log n)$ Depth. In *Proc. of 8th Annual Conference on Theory and Applications of Models of Computation*, pp 164-170.

2011 E. Allender, F. Wang. On the Power of Algebraic Branching Programs of Width Two. In *Proc. of the 38th International Conference on Automata, Languages, and Programming (ICALP)*, pp 736-747.

2011 T. Lee, N. Leonardos, M. Saks, F. Wang. Hellinger Volume and Number-on-the-forehead Communication Complexity. *Manuscript*.