ON LOWER BOUNDS AND PIT FOR PARAMETERIZED ALGEBRAIC MODELS

A THESIS

submitted by

PURNATA GHOSAL

for the award of the degree

of

DOCTOR OF PHILOSOPHY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY MADRAS.

APRIL 2020

THESIS CERTIFICATE

This is to certify that the thesis titled **On Lower Bounds and PIT for Parameterized Algebraic Models**, submitted by **Purnata Ghosal**, to the Indian Institute of Technology, Madras, for the award of the degree of **Doctor of Philosophy**, is a bona fide record of the research work done by her under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. B. V. Raghavendra Rao
Research Guide
Associate Professor
Dept. of Computer Science and Engineering
IIT Madras, 600036

Place: Chennai

Date: 24th April 2020

ACKNOWLEDGEMENTS

I want to begin by thanking my parents, for giving me the opportunity to be at the place in my life I am now. I will always be indebted to the kind and sensitive teachers at Carmel Convent School, Durgapur who shaped my thoughts and were forever encouraging us girls to pursue a career in STEM if we wanted to, despite the disadvantages we faced. I remember the reassuring confidence of my math teacher Ms. Gomathy Kannan in my abilities, which was the only reason I could overcome my anxiety for the school final examination. My friends from the school days, Alorika Chatterjee, Subhra Mazumdar, Deepanjali Sharma were great companions in growing and exploring our futures together.

I was blessed with inquisitive friends in my formative years at IIEST, Shibpur. Prerona Ghosh, Sayak Panja, Sambhuti Pathak were my support system there, and helped me through many difficulties. I found inspiration and companionship in Arpan Das, who found interesting ways to expand the scope of our algorithm designing assignments. His enthusiasm for thinking out of the box made me understand and learn from the courses, and generally led to my interest in pursuing an academic career. At Shibpur I also met Anubrata Das, whose dedication in keeping alive his musical interests made me believe one can find happiness in more than one thing, and that sometimes the joy found in a hobby can immensely help in coping with the stress of research.

At IIT Madras, I am grateful to have attended courses taken by B. V. Raghavendra Rao, my advisor, and Jayalal Sarma, Meghana Nasre and John Augustine. Their teaching made me understand how exciting the subject matters of the courses could be. From their lectures, I also learnt a lot about presenting complicated concepts in a way that is understandable to others. I would specially like to thank Jayalal Sarma for his lectures that stoked much of my interest in computational complexity theory. I will forever cherish the sleepless nights spent by Meenakshi Ray and me in navigating his courses, and in the process discovering our shared interest in the subject.

In the Algorithms and Complexity Theory lab, I met some amazing researchers and

lovely friends in Sumathi Sivasubramaniam, Sajin Koroth, Krishnamoorthy Dinesh, C. Ramya, Subhadra Nanda and Sampriti Roy. I cannot thank Sajin and Dinesh enough for being beside me on my darkest days, whether it was during their stint in IITM or after. They have provided me invaluable advice both academically as my TAs and in research.

Outside the lab, I found excellent company in Sellam Muruganantham, Nikita Saxena, Puja Jha and Rutvi Sanghavi during my initial years at IITM and Sowmya S. Sundaram in the last few years. I thank Rutvi for all the lazy afternoons and nights we spent discussing creative story-lines for possible fiction novels, that made the stress and anxiety of my stint here much more bearable. Sowmya and I have together survived several mental health roller coasters, and I wish her all the best in both her dance and career endeavours in the future. Dr. Sumathi Narayanan guided me through my bad mental health days and provided me tools to handle my anxiety and panic that will last me throughout my life.

Finally, I would like to thank my guide B. V. Raghavendra Rao for painstakingly correcting my mistakes, both in the writing of this thesis and otherwise, and helping through every step of the way in this incredible journey. He has helped me grow as a researcher and as a person and has helped me realise in a fundamental way what a researcher's ethics should be. I hope I am able to put into good use all that I have learnt from him on my way forward.

ABSTRACT

KEYWORDS: Algebraic Complexity Theory, Parameterized Complexity, Lower Bounds, Polynomial Identity Testing, Hitting Set Generators

Polynomials are fundamental objects in mathematics and have wide applications in mathematics as well as computer science. In theoretical computer science, computational problems on graphs are often reduced to computational problems on polynomials. Therefore, a study of polynomials becomes imperative for the progress of research in this area. Algebraic complexity theory is the study of polynomials in terms of the number of elementary arithmetical operations needed for computing a polynomial. In order to count these elementary operations, it is important to represent the polynomial in a succinct manner.

There are several ways of representing a polynomial. The representation of polynomials widely used is the arithmetic circuits model of computation, formally defined by Valiant (1979). An arithmetic circuit of size polynomial in the number of input variables denotes the notion of efficiency in algebraic complexity theory.

Polynomials are grouped into classes on the basis of restrictions on the arithmetic circuit computing them. An important part of the study of algebraic complexity is studying the relationship between classes of polynomials. This question can be formally restated as the arithmetic circuits lower bound problem, which is the main subject of this thesis. The arithmetic circuits lower bound problem is defined as the computation of the minimum size of a circuit in a given class, required for computing a given explicit polynomial.

In this thesis, we study lower bounds on a class of circuits with a semantic restriction of multilinearity. A polynomial is multilinear if the degree of a variable in any monomial cannot exceed one, and a multilinear circuit is such that every gate in the circuit computes a multilinear polynomial. Read-once formulas (ROFs) are a sub-class of multilinear circuits where the underlying directed acyclic graph represents a tree and every input variable is allowed to label only one input node. Multilinear algebraic branching programs are a sub-class of multilinear circuits that can be represented by a layered directed acyclic graph such that every path in this graph computes a multilinear polynomial. Read-once oblivious algebraic branching programs (ROABPs) are a sub-class of multilinear ABPs where input variables are read only once, in a fixed order along every path.

We study the models of sums of ROFs and ROABPs. We obtain a super-polynomial separation between the classes of sum of ROABPs and multilinear ABPs. We also show a lower bound on the size of sum of ROFs computing an explicit polynomial that can be efficiently computed by a ROABP. We show that multilinear ABPs where every sub-program reads variables in intervals i.e., reads variables in an order such that their indices constitute an interval in $[1, n]$ (strict-interval ABPs), are equivalent to ROABPs.

For general arithmetic circuits, Valiant *et al.* (1983) had shown a similar result i.e., a depth of $O(\log n)$ is enough for polynomial size circuits to compute polynomials having polynomial size circuits of unrestricted depth. Later, Agrawal and Vinay (2008) reduced the depth of the desired shallow circuit to a constant 4, with a blow-up in the size. Such results require proving a small upper bound on the size of a shallow circuit for computing any polynomial efficiently computable by circuits of large depth. In the paradigm of parameterized complexity theory, proving upper bounds are simpler since the notion of efficiency is polynomial in terms of the input size, multiplied by any function on an additional parameter $k \ll n$ (known as fixed parameter tractability or FPT). We show that the result of Valiant *et al.* (1983) holds for FPT-size circuits where the parameter is the degree of the polynomial. However, we show that depth-4 circuits parameterized by the degree k where the top product gate fan-in is restricted to $o(k)$, exhibit a lower bound of $n^{O(k)}$ on the size for computing a polynomial that has polynomial size depth-4 degree-parameterized circuits where the top product gate fan-in is unrestricted i.e., $O(k)$. Thus the result of Agrawal and Vinay (2008) cannot be adapted to parameterized arithmetic circuits.

We also obtain parameterized lower bounds on the size of multilinear circuit classes parameterized by degree, for computing an explicit multilinear polynomial of degree k. We construct two explicit multilinear polynomials of degree k, one of which is computable by a depth-4 circuit of FPT size, parameterized by the degree and the other is efficiently computable by a sum of three ROFs. We show $n^{\Omega(t(k))}$ lower bounds, where t is any computable function, on the size of ROABPs, strict-interval ABPs and the model of sum of ROFs with restricted ordering for computing these explicit polynomials. We also show a separation between the class of degree-parameterized read-once and read-2 oblivious ABPs, as read-2 oblivious ABPs can efficiently compute the explicit multilinear polynomials of degree-k.

Finally, we study the computational problem of polynomial identity testing or PIT which is to decide if a given polynomial is identically zero or not. A common tool for obtaining PIT is a polynomial map known as a hitting-set generator. In Ghosal *et al.* (2017), we use a hitting-set generator defined by Shpilka and Volkovich (Shpilka and Volkovich (2009)) along with some other techniques, to obtain PIT for depth-3 circuits parameterized by the degree. We show that the same technique cannot be extended to higher depth circuits parameterized by degree because of the following property: If an n variate polynomial f has a partition of variables such that the partial derivative matrix Raz (2009) has large rank then its image under the Shpilka-Volkovich generator too has large rank of the partial derivative matrix even under a random partition.

We also obtain PIT in the black-box setting using the Shpilka-Volkovich generator for two multilinear classes. One of them is the occur-once formulas with powering gates, which resemble ROFs with powering gates between every two gates on a path in the ROF, and the other is the class of clustered read-2 formulas, which are read-2 formulas such that the lowest level gates computing a read-2 polynomial are a sum of two ROFs.

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVIATIONS

VP	Valiant's P
VNP	Valiant's NP
FPT	Fixed Parameter Tractable
PIT	Polynomial Identity Testing
ACIT	Arithmetic Circuit Identity Testing
ROF	Read-once Formulas
ROP	Read-once Polynomials
ABP	Algebraic Branching Programs
ROABP	Read-once Oblivious Algebraic Branching Programs

NOTATION

$[n]$	$\{1, \ldots, n\}$
X	Set of variables $\{x_1, \ldots, x_n\}$
$\text{var}(f)$	Set of variables occurring in polynomial f
$\text{rank}(M)$	Rank of the matrix M
$\mathsf{rank}_\varphi(f)$	Rank of the partial derivative matrix of f under the partition φ of input variables
$[a, b]$	$\{a, a+1, \ldots, b\}$
$[u, v]_P$	Sub-program of ABP P with source node u, terminal node v
$\langle S \rangle$	The space spanned by the vectors in the set S
$\dim(V)$	Dimension of the vector space V

CHAPTER 1

INTRODUCTION

Multivariate polynomials are fundamental objects in mathematics and play an important role in several areas. Computational problems on polynomials have wide applications in theoretical computer science. For example, polynomials are used to represent combinatorial objects, like graphs, and many results in graph theory are obtained by reducing problems on graphs to problems on polynomials. In the Reed-Solomon coding scheme, polynomials represent encoded messages over a communication channel, and the evaluations of the polynomial on a set of points form a code-word (Reed and Solomon (1960)). Thus, the encoding of messages is reduced to the problem of evaluating a polynomial. Polynomials are also interesting as representations of geometric curves. Hard problems on polynomials can be used in encryption schemes like elliptical curve cryptography (Miller (1986), Koblitz (1987)).

Algebraic complexity theory aims at classifying polynomials based on their complexity. A primary measure of complexity for a multivariate polynomial is the number of primitive algebraic operations (for example, addition and multiplication) required to obtain the polynomial from the input variables and field elements. Arithmetic circuits are used as formal models for computing polynomials, where the size of the circuit represents the number of algebraic operations required to obtain the polynomial. Formally, an arithmetic circuit is defined as a directed acyclic graph having input nodes corresponding to the input variables and constants from the field, internal nodes corresponding to sum and product gates computing the sum and product of their inputs, respectively, and an output gate computing the polynomial represented by the arithmetic circuit. In this representation, the complexity of the polynomial is represented by the size or the number of gates in the circuit and depth or the maximum length of a path from the root node to a leaf node in the circuit. An n-variate degree d polynomial is said to be efficiently computable if it has an arithmetic circuit of size at most polynomial in the number of input variables, n, computing the polynomial. Figure 1.1 shows an arithmetic circuit computing the polynomial $p = (x_1 + x_2)(x_2 + x_3)(x_3 + 6)$.

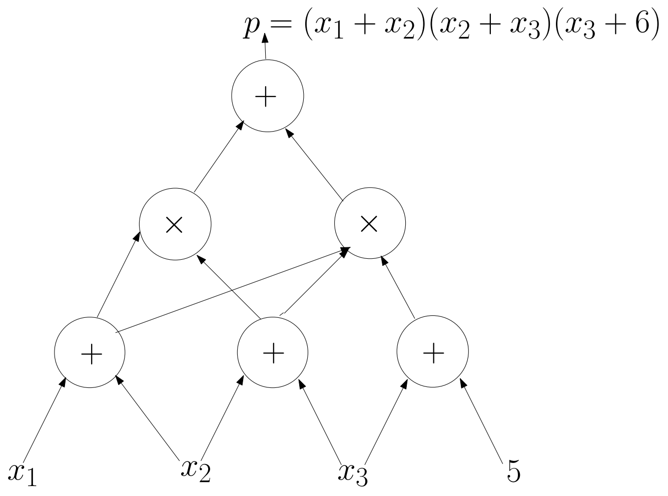


Figure 1.1: Example of an arithmetic circuit.

The families of polynomials, $(p_n)_{n\geq 0}$, of interest are such that for every n, p_n is defined over poly(n) input variables and has degree poly(n). The objective is to classify families of polynomials in to complexity classes, based on the complexity of the arithmetic circuits computing them. Valiant, in his seminal work Valiant (1979), introduced the classes VP and VNP of arithmetic computation, analogous to the well-known boolean complexity classes P and NP. VP is defined as the class of polynomial families that can be computed by arithmetic circuits of polynomial size. VNP is the class of polynomial families $(p_n)_{n\geq 0}$ such that given a monomial in p_n, its coefficient can be efficiently computed (Shpilka and Yehudayoff (2010)).

Polynomial families in VP constitute the notion of efficiently computable in the context of arithmetic circuits. An important member of this class is the family of the determinant polynomial on $n \times n$ symbolic matrices (i.e., matrices where the entries are input variables) defined as

$$\mathsf{det}_n(x_{11}, \dots, x_{nn}) = \sum_{\pi \in S_n} \mathsf{sgn}(\pi) \prod_{i \in [n]} x_{i\pi(i)}, \tag{1.1}$$

where $\mathsf{sgn}(\pi)$ denotes the sign of the permutation π. When the entries of the matrix are field elements, then the determinant polynomial can be efficiently computed using Gaussian elimination. Mahajan and Vinay (1997) give a construction of a polynomial size arithmetic circuit computing the determinant polynomial on a symbolic matrix.

Removing the $\mathsf{sgn}(\pi)$ from the Equation 1.1, the family of permanent polynomials is obtained, which is an important family of polynomials. The family of the permanent polynomial is defined as

$$\mathsf{perm}_n(x_{11},\ldots,x_{nn}) = \sum_{\pi\in S_n}\prod x_{i\pi(i)}. \tag{1.2}$$

Valiant (1979) showed that the permanent polynomial on symbolic matrices of order $n \times n$ is complete for the class VNP. Therefore, the class VNP is also characterised by the family of the permanent polynomial. Henceforth, whenever there is no ambiguity, we will refer to families of polynomials as polynomials.

It is clear from the definitions of the classes that VP $\subseteq$ VNP. Valiant conjectured that the containment of VP in VNP is strict. More specifically, Valiant's hypothesis can be stated as VP $\neq$ VNP (Valiant (1979)).

One approach to Valiant's hypothesis is to prove a lower bound of super-polynomial size on the class of all arithmetic circuits computing the permanent polynomial. This question can be generalized as the arithmetic circuit lower bound problem, which asks the least size required by any circuit from a given class of circuits, $\mathcal{C}$, to compute an explicitly given polynomial that is assumed to be hard.

Arithmetic Circuit Lower Bounds

Proving lower bounds against the class of all arithmetic circuits is one of the fundamental problems on algebraic complexity theory. While a simple counting argument shows that there are polynomials that require exponential size arithmetic circuits, those polynomials are not explicit. The arithmetic circuit lower bound problem is to construct explicit polynomials that cannot be computed by polynomial size arithmetic circuits. Till date, the best known lower bound against general arithmetic circuits is only super-linear i.e., $\Omega(n \log d)$ for computing the polynomial $\sum_{i\in[n]} x_i^{d+1}$ where $d+1$ is the degree of the polynomial (Baur and Strassen (1983)).

Since there has been no improvement in the known lower bounds against general arithmetic circuits, researchers have considered several restrictions on the class of arithmetic circuits. Obtaining lower bounds against smaller, restricted classes may yield techniques that will prove useful in obtaining lower bounds against unrestricted circuits. One such restriction is to consider monotone circuits, i.e., circuits that have no negative constants from the field as inputs. Jerrum and Snir (1982) have shown an exponential lower bound against the size of monotone circuits computing the permanent polynomial. Recently, Yehudayoff (2019) and later, Srinivasan (2019) showed that the monotone versions of the classes VP and VNP could be separated, by proving an exponential lower bound against monotone circuits computing a polynomial that can be efficiently computed by monotone circuits in VNP.

Structural restrictions on circuits seem very natural to consider. Classes of bounded-depth circuits are particularly interesting because of depth reduction of arithmetic circuits. The problem of depth reduction is to compute the minimum depth necessary for an arithmetic circuit of polynomial size such that it has the same computational power as a polynomial size circuit of unbounded depth. Without a restriction on size, every polynomial can be computed by a depth-2 circuit that represents the polynomial as a sum of monomials. When the size of the circuit is restricted to polynomial in the number of inputs, the task of depth reduction gets interesting. Valiant *et al.* (1983) showed that a depth of $O(\log n)$ is enough for computing any polynomial in VP. Later, Agrawal and Vinay (2008) showed a depth reduction to circuits of depth 4 and sub-exponential size.

Circuits of constant depth are defined as follows. For $k > 0$, a depth-k circuit consists of k layers of gates of unbounded fan-in. As the fan-in is unbounded, we assume that the layers alternate between Σ and Π gates. For example, the class $\Sigma\Pi\Sigma$ consists of all families of polynomials computable by polynomial size depth three circuits with the top gate being a Σ gate followed by a layer of Π gates which in turn is followed by a layer of Σ gates.

The depth reduction by Agrawal and Vinay (2008) reduced the depth of a circuit of size $s = \text{poly}(n)$ and degree $d = \text{poly}(n)$ to the constant depth of 4 with sub-exponential size ($2^{o(d+\log \frac{n}{d})} = 2^{o(n)}$ size). This result was built on the depth reduction by Valiant *et al.* (1983) but is not comparable since their construction of the depth-4 circuit results in a sub-exponential blow-up in size, whereas the construction of the shallow circuit by Valiant *et al.* (1983) results only in a polynomial blow-up in size. This blow-up in size was reduced in the construction of depth-4 circuits from general polynomial size circuits given by Koiran (2012), followed by Tavenas (2015), who

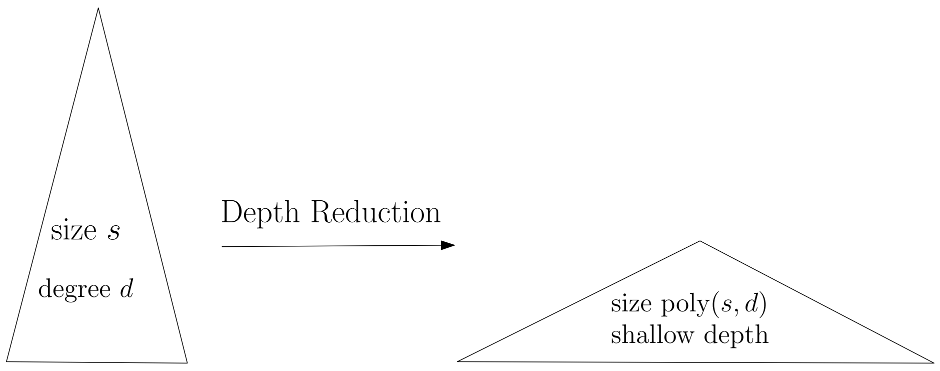


Figure 1.2: General visualisation of depth reduction results.

improved the size of the depth-4 circuit to $n^{O(\sqrt{n})}$ size.

In light of the depth reduction given by Agrawal and Vinay (2008), it is evident that there is a depth-4 circuit of size $n^{O(\sqrt{n})}$ for every polynomial of degree $O(n)$ in VP. Thus, a lower bound of sub-exponential ($n^{\omega(\sqrt{n})}$) size on depth-4 circuits computing the permanent polynomial is sufficient to prove Valiant's hypothesis, since the determinant polynomial is already known to have a polynomial size unbounded-depth arithmetic circuit (i.e., a depth-4 circuit of size at most $n^{O(\sqrt{n})}$) by the result in Mahajan and Vinay (1997). Consequently, the research community shifted focus to the class of bounded depth circuits.

In this regard, Gupta *et al.* (2014) showed a sub-exponential size ($2^{\Omega(\sqrt{n})}$) lower bound against depth-4 circuits computing both the permanent and determinant polynomials. Their work was an important step towards proving the required $2^{\Omega(\sqrt{n}\log n)}$ lower bound against depth-4 circuits computing the permanent polynomial. Kayal *et al.* (2014) showed a lower bound of $n^{\Omega(\sqrt{n})}$ on the size of depth-4 circuits for computing a polynomial defined by Nisan and Wigderson (1997). If the Nisan Wigderson polynomial can be shown to be VNP-complete, a lower bound of $n^{\omega(\sqrt{n})}$ on the size of depth-4 circuits computing it would imply VP $\neq$ VNP, keeping in mind the result of Agrawal and Vinay (2008).

However, immediately afterwards, Fournier *et al.* (2014) showed a lower bound of $n^{\Omega(\sqrt{n})}$ on the size of depth-4 circuits computing the iterated matrix multiplication polynomial using the same proof technique asKayal *et al.* (2014). Since the iterated matrix multiplication polynomial is known to have polynomial size circuits, it is evident that the proof technique would not be useful in separating VP and VNP. In Kumar and Saraf (2015), Kumar and Saraf (2014), the authors obtain super-polynomial lower bounds against more restricted forms of depth-4 circuits.

In order to prove these results, a standard method of proving lower bounds was used. To estimate the size of a circuit in a given class that computes an explicit, hard polynomial, the authors define a complexity measure on polynomials. Subsequently, the measure is shown to take small values for circuits in the given class of circuits and the measure is high for the explicitly given hard polynomial (for example, the permanent polynomial). The well-studied complexity measures are variations of the dimension of partial derivatives measure of the polynomial, defined by Nisan (1991), who used it to obtain lower bounds against non-commutative models. These complexity measures provide useful insight regarding the structure of bounded-depth circuits and are also useful in proving lower bounds against smaller classes of circuits, which we encounter in this thesis.

Motivation: Multilinear Models

Since the best known lower bound for general arithmetic circuits is unsatisfactory, and the lower bounds on the size of constant-depth circuits have not been useful for proving Valiant's hypothesis, authors have considered other restrictions on arithmetic circuits (Raz (2006), Raz and Yehudayoff (2008)).

While most widely studied restrictions such as bounded depth circuits are structural in nature, researchers have also considered semantic restrictions such as homogeneous circuits and multilinear circuits. Homogeneous circuits are circuits where all inputs to a product gate are polynomials of the same degree. However, since it is possible to build a homogeneous circuit from a non-homogeneous circuit (i.e., homogenize the circuit) with only a polynomial blow-up in size, homogeneity does not yield interesting, special cases.

In a multilinear circuit, every gate is assumed to compute a multilinear polynomial i.e., a polynomial where every input variable has individual degree one. Multilinear circuits are a natural model for computing multilinear polynomials. As the determinant and permanent polynomials are multilinear polynomials, it is important to study Valiant's hypothesis on multilinear circuits.

An important sub-class of multilinear circuits is the class of *syntactically multi-*

linear circuits. In a syntactically multilinear circuit, inputs to every product gate are polynomials on disjoint sets of variables. *Syntactically multilinear formulas* are a subclass of syntactically multilinear circuits where the underlying directed acyclic graph is a tree. Raz, in his seminal work Raz (2009) showed a super-polynomial lower bound against the class of syntactically multilinear formulas computing the permanent and determinant polynomials. Consequently, Raz (2006) showed a separation between the classes of multilinear circuits and syntactically multilinear formulas. Raz and Yehudayoff (2008) defined explicit multilinear polynomials and showed size lower bounds against syntactically multilinear formulas computing both these polynomials.

Algebraic branching programs (ABPs) are a model of computation of polynomials where the underlying graph is a layered directed acyclic graph. There is a source and a terminal node in the first and last layers and the intermediate layers contain nodes that compute the sum of their inputs. Edges between consecutive layers are labelled by input variables or constants from the field. Every source to terminal path computes a polynomial that is a product of the labels of the edges in the path. The polynomial computed at the terminal node is the sum of all the polynomials computed by the source-to-terminal paths. Multilinear ABPs are ABPs where every path in the ABP computes a multilinear polynomial. Dvir *et al.* (2012) showed a super-polynomial separation between multilinear ABPs and syntactically multilinear formulas.

Further restrictions are applied on multilinear circuits in the quest for good lower bounds, such that these restrictions provide insight on the source of complexity of multilinear circuits. One such restriction is on the number of reads of variables in the formula i.e., restricting the number of times a variable can occur as a label. While in a circuit, it can be assumed without loss of generality that a variable occurs at most once as a label, in the case of a formula, allowing only a bounded number of reads is a restriction.

Formulas where a variable can occur at most once are known as read-once formulas (ROFs). The class of ROFs has received wide attention in the literature (Shpilka and Volkovich (2015)). A multilinear polynomial computed by a ROF is called a read-once polynomial (ROP). A ROF has size at most $O(n)$ as at most n leaves can be labelled with variables. Thus there is no question of lower bounds against ROFs. However, any multilinear monomial is a ROP and hence any multilinear polynomial can be expressed as sum of ROPs. Ramya and Rao (2019*b*) considered the sum of ROPs as a computational model and proved an exponential lower bound for sum of ROPs computing a polynomial in VP defined by Raz and Yehudayoff (2008).

Read-once Oblivious ABPs (ROABPs) are layered ABPs such that every variable labels an edge only once along a path in the ABP, and the edges between two consecutive layers are all labelled by exactly one variable. Nisan (1991) proved an exponential lower bound against the size of a ROABP computing the palindrome polynomial. More recently, Kayal *et al.* (2016) showed a separation between ROABPs and depth-3 multilinear circuits by defining a hard polynomial in each class that could not be efficiently computed by the other. Ramya and Rao (2018) obtained an exponential lower bound on the size of the sum of ROABPs computing the Raz-Yehudayoff polynomial.

In this thesis, we study different multilinear classes and prove lower bounds against such classes computing the explicit polynomials defined by Raz and Yehudayoff (2008) and Dvir *et al.* (2012).

Motivation: Parameterized Arithmetic Circuits

From the above discussion on lower bounds, it is clear that there has been notable progress on proving lower bounds on special classes of arithmetic circuits, but the problem still remains open for unrestricted arithmetic circuits, for which no easy strategy seems to be available so far. This motivates the necessity of looking into other algorithmic paradigms, like the parameterized paradigm, in search of useful tools. Parameterized complexity, defined by Downey and Fellows (1999), is the study of computational problems where the complexity of problems is expressed in terms of both the input size n and an arbitrary function of an additional parameter k such that the parameter is considered to be independent of the input size, $k \ll n$. The notion of efficiency in the parameterized setting, known as *fixed parameter tractability* is characterised by parameterized algorithms that run in $g(k)\text{poly}(n)$ time, g being an arbitrary computable function. Henceforth, we will refer to fixed parameter tractability as *FPT* in this thesis.

The motivation for considering the parameterized paradigm for algebraic computation arises from graph theory. Koutis and Williams (2009) and Williams (2009) show that detecting a tree or a path of a given constant length k can be reduced to detecting the existence of a multilinear monomial in a homogeneous polynomial of degree k defined on the graph such that the degree of the polynomial is exactly k. These ideas were further extended in Amini *et al.* (2012) to compute the number of proper k-colorings of a graph. From such results, it is clear that reducing the problem of detecting a graph theoretic property in a given graph to solving a computational problem on the a polynomial defined from the graph is an efficient and commonly used strategy. Since the degree of the polynomial, k, is equal to the size of the required structure (i.e., length of a path or size of a tree) in all these problems, this gives an immediate connection to an efficient parameterized algorithm for these problems with k as the parameter.

From the perspective of algebraic complexity theory, we ask if depth reduction results given by Valiant *et al.* (1983) and Agrawal and Vinay (2008) hold under parameterization. With the allowance of an arbitrary function of the parameter in the complexity, it is easier to prove an upper bound on the size of a small-depth circuit which would have the same computational power of a circuit with FPT size and unrestricted depth. However, we observe that the result of Agrawal and Vinay (2008) does not hold under parameterization by degree, though the reduction of depth to $O(\log n)$ given by Valiant *et al.* (1983) still holds.

It is also interesting to investigate if known lower bounds can be strengthened or generalized by expressing the lower bound problem as parameterized by the degree of the polynomial. Many lower bounds known in the classical setting, like the result by Raz (2009), do not yield a large (i.e., greater than FPT) lower bound in the parameterized setting. Hence coming up with explicit constructions of hard polynomials parameterized by degree and lower bounds against parameterized classes computing such a hard polynomial is an interesting task in itself.

Parameterized Computational Problems

Since we have studied parameterized lower bounds, it is interesting to study other computational problems parameterized by the degree in order to get a comprehensive view of the parameterized algebraic complexity landscape. A natural computational problem on polynomials is the polynomial equivalence problem, which asks, given two polynomials, whether one is equal to the other. A polynomial can be given as an input in multiple ways: it can be given as a black-box, such that, given an assignment to the n

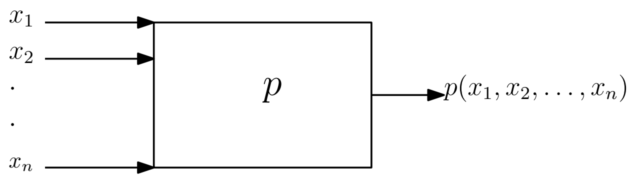


Figure 1.3: Polynomial p represented as a black-box.

variables, the black-box for a polynomial p gives the evaluation of p on the assignment. Or the polynomial can be given in the form of an arithmetic circuit computing it, in which case, the output of every intermediate gate is also known (also called white-box representation). When these two polynomials are given using the same representation, this problem is the same as asking whether the difference between the two polynomials is the zero polynomial, which is known as *polynomial identity testing* (PIT).

An important motivation for our study of PIT has been the result by Kabanets and Impagliazzo (2004) that states that an efficient algorithm for PIT on polynomials represented in the black-box form implies lower bounds for arithmetic circuits. Thus these computational problems central to algebraic complexity theory, are also connected to each other.

An efficient randomized algorithm for PIT in the black-box setting was independently given by authors hailing from varied scientific perspectives of the problem, starting with the mathematician Ore (1922) and followed by computer scientists Schwartz (1980), Zippel (1979) and DeMillo and Lipton (1978) respectively. However, without the help of randomness, the PIT problem could be efficiently solved only for very restricted classes of circuits.

Constant-depth circuits, as seen earlier, are defined such that the top output gate is always a sum gate. This is because PIT of a depth-k circuit with a top product gate reduces to solving PIT on its depth-$(k-1)$ factors i.e., detecting if at least one factor is a zero polynomial. PIT for depth-3 circuits have been widely studied, where Kayal and Saxena (2007) gave an efficient algorithm in the white-box setting and efficient black-box PIT is known only when the top fan-in is bounded, where the increasingly improved bounds have been given by Kayal and Saraf (2009) and later Saxena and Seshadhri (2010). In fact these results were the precursor to the discovery of the all-powerful nature of depth-4 circuits via depth-reduction of circuits. Though there have been a lot of PIT results on smaller, notably multilinear models, efficient black-box PIT for depth-3 and depth-4 circuits is still unknown.

In the context of parameterized complexity, Chauhan and Rao (2015) gave a degree-parameterized version of the randomized black-box PIT algorithm given by Schwartz (1980) and Zippel (1979). The authors reduced the complexity of the algorithm such that it could be derandomized in FPT-time. Since constant-depth circuits are important due to existing depth-reduction results and are well-studied for PIT , in Ghosal *et al.* (2017) we obtain a FPT-time depth-3 PIT for white-box representation of the polynomial. The simplicity of the algorithm compared to the white-box depth-3 PIT algorithm in the non-parameterized setting, motivates us to explore parameterized PIT and see if our techniques for the depth-3 case can be extended to higher depth circuits.

Contributions of this Thesis

In this section, based on the directions mentioned in the previous sections, we go through our results in this thesis. Motivated by the recent lower bounds on multilinear classes like sum of ROFs (Ramya and Rao (2019*b*)), ROABPs and strict-interval ABPs (Ramya and Rao (2019*a*)), we continue their study of lower bounds on multilinear models on other multilinear circuit classes in the non-parameterized setting (Chapter 3). The classes we consider are restricted in number of times a circuit in the class can read the variables and the order in which the variables can be read. The hard polynomials we consider are an explicit multilinear polynomial defined by Dvir *et al.* (2012) and a variation of the explicit polynomial defined by Raz and Yehudayoff (2008).

Both of these polynomials attain full rank of the partial derivative matrix, a complexity measure that is a variation of the rank of partial derivative matrix given by Nisan (1991), defined by Raz (2006). However, the polynomial defined by Dvir *et al.* (2012) attains full rank only against a special class of partition functions denoted by *arc-partitions*, and the rank of the partial derivative matrix against these partitions is defined as the *arc-rank*. In our first result, we use the standard lower bound strategy to show that if a sum of s ROABPs is able to compute the DMPY polynomial, then s must be large, in this case, sub-exponential in n.

Theorem 1. *Let $f = f_1 + \ldots + f_s$, each $f_i \in \mathbb{F}[X]$ being computable by a ROABP of size poly(n). If f is the full arc-rank polynomial f_D defined by Dvir* et al. *(2012) then*

$s = \mathit{exp}(\omega(n))$.

Our proof uses the techniques developed by Dvir *et al.* (2012) to analyse the effect of a random arc-partition on a ROABP.

However, it is not clear if the techniques developed by Raz and Yehudayoff (2008) and Dvir *et al.* (2012) can be applied to separate further restricted models. For example, it is not clear if it is possible to separate the class of sum of ROFs from the class of polynomial size ROABPs.

For such a result, one needs a distribution $\mathcal{D}$ on partitions of the input variables such that a full rank polynomial on all partitions in the support of the distribution is efficiently computable by a ROABP and any ROF has low rank under a random partition from $\mathcal{D}$. To this end, we construct a polynomial f_{PRY} computable by ROABPs such that any sum of ROFs computing it requires exponential size.

Theorem 6. *Let $f_1, \dots, f_s$ be read-once polynomials such that $f_{\mathsf{PRY}} = f_1 + f_2 + \cdots + f_s$, then $s = 2^{\Omega(n)}$.*

Thus our results, along with the lower bound obtained by Dvir *et al.* (2012), illustrate the difference in power among the multilinear models of syntactically multilinear formulas, ROABPs and read-once formulas.

The models we have considered in our lower bound results are restricted to read the input variables only once. Other models that we often consider are classes of multilinear circuits that read variables in an order such that the indices of variables read by any gate in the circuit form an interval $[i, j],\ i < j$. Consider a ROABP P where variables occur in the order $x_1, \dots, x_n$. For any two nodes $u,\ v$, the sub-program of P with u as the source node and v as the terminal node computes a polynomial in the variables $\{x_i, x_{i+1}, \dots, x_j\}$ for some $i \le j$. In other words, for every $u,\ v$ we can associate an interval $[i, j],\ i \le j$ of $[1, n]$. Generalizing ROABPs by using this interval property, Ramya and Rao (2019*a*) introduced the model of strict-interval ABPs (see Definition 4 for a formal definition).

Strict-interval ABPs are a sub-class of the interval ABPs model defined and used by Arvind and Raja (2016), who showed a lower bound against the model. In Chapter 3, we show that strict-interval ABPs do not have more computational power compared to ROABPs, i.e.,

Theorem 5. *The class of strict interval ABPs are equivalent to the class of ROABPs.*

As the results obtained in Chapter 3 illustrate lower bounds on very restricted classes of circuits, and since there has been no progress on lower bound results on general circuits in recent times, we are interested in exploring a parameterized perspective of arithmetic circuits. In Chapter 4 we study parameterized depth-reduction of algebraic circuits parameterized by the degree. Our motivation is that since the relaxed notion of efficiency in the parameterized paradigm results in the ease of obtaining upper bounds, it might be possible to obtain better depth-reduction results. We inspect if the seminal depth-reduction by Valiant *et al.* (1983) to depth $O(\log n)$ holds for circuits parameterized by degree. On adding the factor contributed by the function on the parameter to the complexity of circuits considered in the depth reduction by Valiant *et al.* (1983), we arrive at the following statement:

Any parameterized polynomial family (p, k) in FPT can be computed by circuits of depth $g(k) \log n$ and size $g'(k)n^{O(1)}$, for some functions g and g' that depend only on the parameter.

The proof of this statement would follow from the proof of the original statement itself. A natural next step would be to obtain a parameterized analogue of the depth reduction of circuits to constant-depth, given by Agrawal and Vinay (2008), in the parameterized context i.e., given a circuit of size $g(k) \log n$ and unbounded depth computing a polynomial p of degree k, if it possible to construct a depth 4 circuit with a reasonable blow-up in size, computing the same polynomial p. We show that such a statement does not hold true for the restriction of product gate fan-ins being bounded.

Corollary 2. *There is a parameterized family of polynomials $(p_n)_{n\geq 0}$ that can be computed by $\Sigma\Pi^{O(k)}\Sigma\Pi^{k}$ circuits of FPT size, but any depth four $\Sigma\Pi^{o(k)}\Sigma\Pi^{k}$ circuit computing it requires size $n^{\Omega(k)}$.*

In case of degree-parameterized depth-4 circuits, we show a lower bound against circuits of top product gate fan-in $o(k)$ computing an explicit polynomial that has FPT size depth-4 circuit of unrestricted top product gate fan-in. Since depth-4 circuits where the top product gate fan-in is restricted to $o(k)$ have less computational power than depth-4 circuits with top product gate fan-in $O(k)$, a depth reduction of circuits to depth-4 is not possible. Thus the result of Agrawal and Vinay (2008) does not hold for depth-4

degree-parameterized circuits. However we don't know if we can obtain an analogous depth reduction result for some other constant depth larger than 4, or if a depth reduction to depth-4 circuits is possible with a different parameterization.

Having explored the possibility of a parameterized depth reduction and studied parameterized constant-depth circuits, we follow the direction taken by algebraic complexity literature in the non-parameterized setting to proceed to investigate parameterized lower bounds against multilinear algebraic models. In Chapter 5 we define two multilinear polynomials parameterized by degree, that have high rank under a random partition. The first polynomial f attains the notion of full rank in the parameterized setting, $g(k)n^{k/2}$ (where g is a computable function).

The design of the polynomial f is based on perfect matchings in the complete graph on $2k$ vertices, $K_{2k} = (V, E)$. An edge polynomial p_{uv} is defined for each edge $(u, v) \in E$. A polynomial resembling the product of k edge polynomials for the edges in a perfect matching M represents a degree-$2k$ matching polynomial p_M. The hard polynomial f is formally represented as a parameterized family of polynomials $f = (f_{n,2k})_{n>1,2k|n}$, $f_{n,2k} \in \mathbb{G}[x_1, x_2, \ldots, x_n]$ as follows:

$$f(x_1, x_2, \ldots, x_n) = \sum_{M \in \mathcal{M}} \zeta_M \prod_{(i,j)\sim M} (1 + p_{ij}(V_i \cup V_j)),$$

where ζ_M are formal variables that are assigned $0/1$ values such that the polynomial f attains close to full rank of the partial derivative matrix. The edge polynomial p_{ij} is a n/k-variate quadratic multilinear polynomial defined as

$$p_{ij}(x_i, \ldots, x_{j+n/2k-1}) = \sum_{k<\ell} \omega_{k,\ell} x_k x_\ell.$$

Here $\omega_{i,j}$, for $1 \le i < j \le n/k$, are also formal variables. Formally, we have:

Theorem 8. *For the parameterized multilinear polynomial family $f = (f_{n,2k})_{n,k\geq 0}$, we show,*

$$\mathsf{rank}_\varphi(f_{n,2k}) = \Omega\left(\frac{n^k}{(2k)^{2k}}\right),$$

for every equi-partition $\varphi : X \to Y \cup Z$ and $k > 3$.

Here, $\mathsf{rank}_\varphi(f)$ is a complexity measure for the polynomial f (see Section 2.4.1

for a definition). From the definition of f, it follows that for $n,\ k > 0$, $f_{n,2k}$ can be computed by a $\Sigma\Pi^k\Sigma\Pi^2$ circuit of size $k^{O(k)}n^O(1)$.

It is also interesting to investigate if hard polynomials can be constructed from restricted multilinear circuits. Additionally, in view of our interest in construction of hard polynomials based on perfect matchings, we consider the polynomial defined from the union of 3 perfect matchings by Kayal *et al.* (2016), of degree $O(n)$. We were able to construct a family of degree-parameterized explicit multilinear polynomial based on the construction of Kayal *et al.* (2016), which we denote by $h_{n,k}$ for $n,\ k > 0$. The polynomial $h_{n,k}$ also attains close to full rank, and is formally defined as follows.

$$h_{n,k}(x_1,\ldots,x_n) = \sum_{i\in[3]} w_i \left(\prod_{j\in[\frac{k}{2}]} \sum_{(u,v)\in B_{ij}} x_u x_v \right), \tag{1.3}$$

where w_i are formal variables substituted with $0/1$ values to choose one of the three matching polynomials. The exact value of the rank measure is illustrated in the following result. When $n,\ k$ are clear from the context, we denote $h_{n,k}$ by h.

Theorem 9. *Let h be the polynomial defined in Equation 1.3. Then there is a constant $c = \frac{23+20\epsilon}{114}$ for a fixed $\epsilon > 0$ such that for every equi-partition $\varphi : X \to Y \cup Z$, over the rational function field $\mathbb{F}(w_1, w_2, w_3)$ such that,*

$$\mathsf{rank}_\varphi(h) \geq \left(\frac{n}{k}\right)^{ck}.$$

This polynomial can be efficiently computed by a sum of three ROFs. We note that in the parameterized setting, the full value of the complexity measure is easily achieved by polynomials computable in an efficient manner by relatively simple i.e., restricted models, unlike in the non-parameterized case. Hence it is comparatively more difficult to obtain good (i.e., $n^{O(k)}$) lower bounds in the parameterized setting than obtaining exponential lower bounds in the classical setting.

In this chapter, we obtain a $n^{O(k)}$ lower bound against the size of a ROABP computing the polynomial f.

Theorem 10. *A ROABP P computing the polynomial family $f = (f_{n,2k})$ requires size*

$$S = \Omega(n^k/(2k)^{2k}).$$

A similar lower bound is obtained against the size of a ROABP computing f. Thus the class of sum of three ROFs contain polynomials that cannot be efficiently computed by a ROABP, whereas ROFs are notably weaker than ROABPs.

Theorem 11. *An ROABP computing the family of polynomials $(h_{n,k})_{n,\ k>0}$ defined in Equation 1.3 requires size $n^{\Omega(k)}$.*

A polynomial computed by a parameterized ROABP of FPT size has a FPT sized upper bound on the complexity measure, thus yielding strong parameterized lower bounds. However, with only one more read of the variables allowed in the same order, the complexity increases exponentially.

Theorem 12. *The family of polynomials $(h_{n,k})_{n,\ k>0}$ defined in Equation 1.3 can be computed by read-2 oblivious ABPs of size $n^{O(1)}$.*

This gives a wide separation between the parameterized classes of read-once and read-twice ABPs. This is not surprising since h, a sum of 3 ROFs, attains nearly full rank, hence ABPs being a stronger model than formulas, should naturally attain full rank in lesser number of reads.

Corollary 3. *There is a parameterized polynomial family computable by polynomial size read-2 ABPs such that any ROABP computing it has size $n^{\Omega(k)}$ where k is the parameter.*

We show a smaller lower bound than ROABPs on the size of parameterized strict-interval ABPs computing f. The proof proceeds by expressing a strict-interval ABP in the form of a very structured depth-4 parameterized circuit for which the upper bound on the complexity measure is small.

We also obtain a good lower bound against the class of sum of ROFs, where the order in which variables are read is fixed. We state the result formally as follows:

Theorem 14. *Let $p_1, \ldots, p_s$ be ROFs where variables are read in a particular order, such that $f = p_1 + \cdots + p_s$. Then, $s = (n^{\Omega(k)}/t(k))$, where t is a computable function on k.*

We show this by building a graph corresponding to each ROF. We upper bound the rank measure on each ROF by counting edges in the graph that correspond to monomials in the polynomial that would contribute to high rank. The product of s with this upper bound gives the upper bound on rank for the sum of ROFs class. As this upper bound must equal the rank lower bound on f, we obtain a lower bound on the size of the sum, s.

The investigation on parameterized lower bounds motivates us to examine parameterized PIT, since the two problems are related by a result of Kabanets and Impagliazzo (2004). It is a common technique to use a function known as hitting-set generator to easily obtain a set of witnesses for the polynomial being non-zero. We consider one such hitting-set generator defined by Shpilka and Volkovich (2008). This hitting-set generator yielded a very simple white-box PIT algorithm for depth-3 circuits (Ghosal *et al.* (2017)) and is natural and convenient to work with in the parameterized setting. This is because it transforms any n-variate polynomial of degree k into a k-variate degree-nk polynomial that is identically zero if and only if the original polynomial is identically zero, which makes it possible to evaluate the k-variate polynomial on all possible assignments in FPT-time to ascertain whether it is a zero polynomial. In Chapter 6, we extensively study this generator, which we denote by the term *SV-generator*.

Minahan and Volkovich (2018) gave an elegant characterisation for the SV-generator G_k acting on a polynomial P which helps in analysing $G_k(P)$, thus helping in designing black-box PIT algorithms. They obtained efficient black-box PIT for the sum of ROFs using this expression. We use this expression to obtain black-box PIT for two restricted non-parameterized multilinear models that we define. The first model is that of a modified read-once formula, where we have arbitrary powering gates between any two gates (sum or product) in the circuit. We define this model *occur-once formula with powering gates*.

Corollary 5. *Let $f \in \mathbb{F}[x_1, \ldots, x_n]$ be a polynomial computed by a polynomial-sized occur-once formula with powering gates with maximum fan-in p, f is non-constant. Then $G_1(f)$ is also non-constant i.e., a hitting-set generator for f.*

We also define the model of *clustered read-2 formulas*, which is formally defined in Definition 31. We show:

Theorem 18. *G_7 is a hitting-set generator for the class of clustered read-2 formulas.*

Having observed the effectiveness of the SV-generator for a variety of models, we aimed to extend the depth-3 parameterized white-box PIT to higher depths as a natural next step. Depth-4 circuits parameterized by degree have relatively higher value of the rank complexity measure than depth-3. Even obtaining efficient white-box PIT is difficult for depth-4 circuits, since depth-4 circuits of $n^{O(\sqrt{n})}$ size are computationally equivalent to unrestricted polynomial size circuits (Tavenas (2015)). A white-box PIT is not evident using existing results even for the k-variate polynomial $G_k(f)$, where f is a polynomial having a FPT-size depth-4 parameterized circuit, G_k being the SV-generator. We observe the reason for this being the SV-generator preserves the rank complexity measure of a given polynomial P i.e., the polynomial $G_k(P)$ has nearly the same value of the rank measure as P. We express this result formally in the following theorem.

Theorem 19. *Let $f \in \mathbb{F}[x_1, \dots, x_n]$ be a polynomial of degree $\leq k$. Let $g = G_{2k}(f)$. Then,*

$$\exists \varphi, \mathsf{rank}(M_{f^\varphi}) \geq R \implies \mathbf{Pr}_{\varphi'}[\mathsf{rank}(M_{g^{\varphi'}}) = R] \geq \Omega(1/2^{2k}),$$

where the probability is taken over the uniform distribution over the set of all partitions of the variables in g into two parts of equal size.

Theorem 19 can be extended to a more general class of hitting set generators, which we denote by *SV-like* generators. We prove the rank-preserving property for SV-like generators as follows.

Corollary 6. *Let $H = (H_{n,2t})_{1 \leq 2t \leq n}$ be an SV like hitting set generator such that $H_{n,2t}$ is a hitting set generator for degree t polynomials on n variables. Let $f = f(x_1, \dots, x_n)$ be any polynomial of degree $t/2$ and $h = H(f)$. Then,*

$$\exists \varphi, \mathsf{rank}(M_{f^\varphi}) \geq R \implies \exists \varphi' \mathsf{rank}(M_{h^{\varphi'}}) \geq R.$$

However, this does not rule out other families of hitting set generators from having the rank-preserving property, neither does it rule out the possibility of depth-4 parameterized white-box PIT via other hitting set generators.

Thus our study covers the two fundamental problems of lower bounds and PIT in algebraic complexity theory and examines the insight provided by parameterized algebraic complexity theory regarding the structures of these problems. Parameterization of a problem generalizes it since the complexity can be in terms of any function of the parameter. Hence, we examine problems in algebraic complexity through a wider lens provided by the fixed parameter of degree, and illustrate the limitations of the techniques frequently used in the non-parameterized scenario in our parameterized world.

Chapter-wise Overview

- In Chapter 3 we obtain a sub-exponential separation between multilinear ABPs and ROABPs (Theorem 1) followed by the equivalence of the classes of strict-interval ABP and ROABP (Theorem 5) and an exponential separation between the classes of sum of ROFs and ROABPs (Theorem 6).
- In Chapter 4 we show that the depth reduction by Valiant *et al.* (1983) holds even for circuits where the product gate fan-ins are bounded by a function of the degree of the polynomial (Proposition 5) but the depth reduction to depth-4 circuits by Agrawal and Vinay (2008) does not hold in this setting (Corollary 2).
- In Chapter 5 we obtain parameterized lower bounds against multilinear parameterized classes computing two explicit polynomials, one of which is efficiently computable by a parameterized depth-4 circuit (Theorem 8) and the other is a sum of three ROFs parameterized by the degree (Theorem 9).
- In Chapter 6 we show the application of SV-generator for obtaining black-box PIT for occur-once formulas with powering gates (Theorem 16) and clustered read-2 formulas (Theorem 18). We mainly prove a rank preserving property of the SV-generator (Theorem 19) and SV-like generators (Corollary 6) because of which SV-generator cannot be used to yield PIT for constant-depth circuits of depth larger than 3, parameterized by degree.

CHAPTER 2

PRELIMINARIES

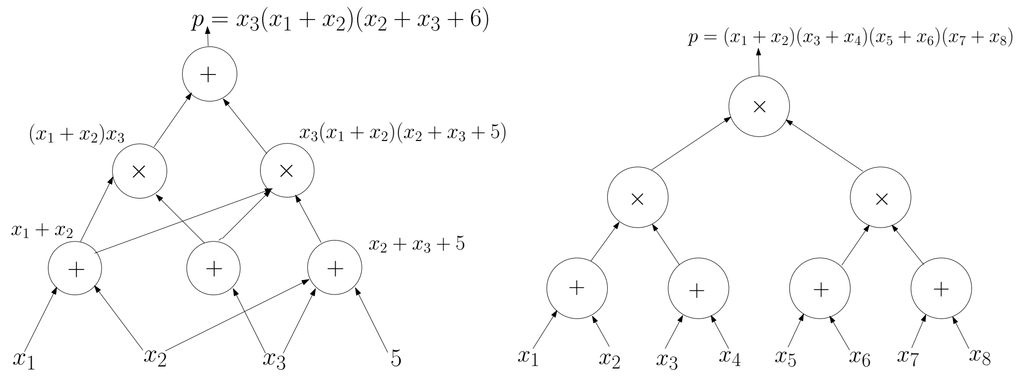
In the following sections we establish all concepts and notations used in this document to explain our results. We begin with a description of the models of computation used to compute or represent polynomials, followed by basic definitions and concepts used in the paradigm of parameterized complexity and the different measures of complexity used in this document to show the relations between the different models of computation. For a detailed description, the reader is referred to the surveys Shpilka and Yehudayoff (2010) and Saptharishi *et al.* (2016) and Downey and Fellows (1999) for the basic concepts of parameterized complexity theory.

2.1 Algebraic Computational Models

In this section we describe arithmetic circuits, that represent polynomials $p(x_1, \ldots, x_n)$ over the polynomial ring $\mathbb{F}[x_1, \ldots, x_n]$ succinctly, and the complexity measures associated with them. We denote the set of variables $\{x_1, \ldots, x_n\}$ by X, degree of a polynomial p by deg(p) and the set of variables that are inputs to p by var(p).

Arithmetic Circuits An arithmetic circuit is a directed acyclic graph (DAG in short) with input nodes of in-degree zero labelled by variables in X and constants from the field $\mathbb{F}$, internal nodes and an output node of out-degree zero, called *gates*, labelled by $+$ or $\times$. Every internal node g in the circuit computes a polynomial which is either the sum or the product of its inputs, according to its label. The arithmetic circuit is said to be computing the polynomial p if the polynomial computed by the output node is p.

The complexity measures associated with an arithmetic circuit C are size(C), which may denote either the number of gates or the number of edges in the circuit, and depth(C) which denotes the length of the longest path from any input node to the output node in the circuit. Figure 2.1(a) shows an arithmetic circuit of size 10 and depth 3.


$p = x_3(x_1+x_2)(x_2+x_3+6)$
$(x_1+x_2)x_3$
$x_3(x_1+x_2)(x_2+x_3+5)$
x_1+x_2
x_2+x_3+5
x_1
x_2
x_3
5
$p = (x_1+x_2)(x_3+x_4)(x_5+x_6)(x_7+x_8)$
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8

(a) Arithmetic circuit C (b) Arithmetic formula F

Figure 2.1: (a) An arithmetic circuit C with the polynomials computed by each gate visualised, followed by (b) an example of an arithmetic formula F.

A class $\mathcal{C}$ of circuits is a collection of arithmetic circuits that usually share a common property. In this section we will go through the definitions of the well-studied special classes of arithmetic circuits and describe their computational power with respect to class of all arithmetic circuits. Circuits of constant depth have been widely studied in the literature Saptharishi *et al.* (2016). Though we can assume every gate in a general arithmetic circuit has a constant fan-in (say, fan-in 2), we assume gates in a circuit C of constant depth have unbounded fan-in, since otherwise, C cannot read all the input variables.

Arithmetic circuits where the underlying DAG is a tree are known as *arithmetic formulas*. Figure 2.1(b) shows an arithmetic formula computing the polynomial $p = \prod_{i=1}^{4}(x_{2i-1}+x_{2i})$.

Multilinear circuits A monomial in the polynomial p is denoted by $x_1^{\alpha_1} \cdot \ldots \cdot x_n^{\alpha_n}$, where $\forall i, x_i \in X$ and $\alpha_i \in \mathbb{F}$ are constants that denote the individual degree of x_is. A monomial is multilinear when $\forall i, \alpha_i$ is either 1 or 0. The polynomial p is multilinear if all its monomials that have non-zero coefficients are multilinear. An arithmetic circuit C computing p is a multilinear circuit if every gate in the circuit computes a multilinear polynomial. It is not known whether every multilinear polynomial that can be computed by a polynomial sized circuit, has a polynomial sized multilinear circuit computing it. Though multilinearity is a semantic restriction on the circuit, a variant of multilinearity, *syntactic multilinearity* has received wide attention as a syntactic restriction on circuits.

A circuit is said to be *syntactically multilinear* if the inputs to every product gate in the circuit are defined on disjoint sets of variables i.e., if a product gate f is such that $f = g \times h$, g, h being inputs to the gate f, then $\text{var}(g) \cap \text{var}(h) = \emptyset$.

We are interested in several restricted classes of arithmetic formulas, especially sub-classes of syntactically multilinear formulas like read-once formulas (ROF in short) and interval formulas. We define both of these models here.

Definition 1. (Read-once formulas) A syntactically multilinear arithmetic formula is a read-once formula if every input variable labels an input node exactly once.

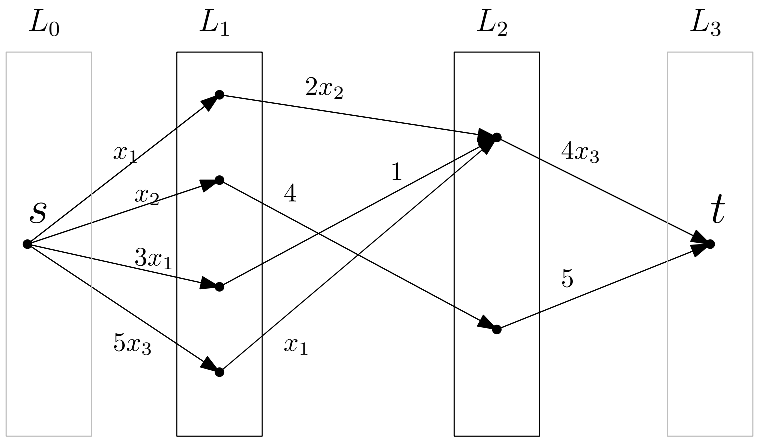
Let g be a gate in a read-once formula F such that $g = g_1 + g_2$ or $g = g_1 \times g_2$, where g_1 and g_2 are its inputs. Then the polynomials computed by g_1 and g_2 are defined on disjoint sets of variables.

Polynomials computed by ROFs are known as read-once polynomials (ROPs). ROFs are a proper sub-class of syntactically multilinear formulas.

Read-once formulas can be generalized to *read-k formulas* where every input variable is read k times i.e., each variable labels at most k input nodes, for a $k > 0$.

A boolean function $f : \{0, 1\}^n \to \{0, 1\}$ is often represented by a *branching program* P, which is a layered DAGs such that every path represents a full or partial assignment to the inputs of f, and the paths either end at the terminal node corresponding to $f(x_1, \ldots, x_n) = 0$ or the terminal node corresponding to $f(x_1, \ldots, x_n) = 1$. *Algebraic branching programs* (ABPs in short) are algebraic analogues of the boolean model of computation of branching programs. It is known that the class of ABPs subsumes the class of arithmetic formulas. ABPs and their various interesting sub-classes are formally defined as below.

Algebraic branching programs An algebraic branching program (ABP) P computing a polynomial is a layered DAG with each layer containing nodes and edges between nodes in any two consecutive layers labelled by variables in X and constants from the field $\mathbb{F}$. Let the layers be $L_1, \ldots, L_r$. Then L_1, the first layer, contains only the source node s that has zero in-degree and L_r, the last layer, contains only the terminal node t. Every path ρ from s to t computes a polynomial $p_\rho = \prod_{e \in \rho} \text{label}(e)$, where e is an edge and $\text{label}(e)$ denotes the constant or variable labelling e. Every sub-program between two nodes u, v in P is denoted by $[u, v]_P = \sum_\rho p_\rho$ where ρ is a path from u to v and



$p = 8x_1x_2x_3 + 20x_3^2x_1 + 12x_1x_3 + 20x_2$

Figure 2.2: Example of an algebraic branching program.

p_ρ is the polynomial computed by the path ρ. The polynomial computed by P is $[s,t]_P$. Figure 2.2 illustrates an example of an ABP.

Complexity measures associated with an ABP P are size(P), which denotes the number of nodes in P, depth(P) which denotes the number of layers in P and width(P) which denotes the maximum number of nodes in any layer in P.

We can define syntactically multilinear ABPs as follows:

Definition 2. A syntactically multilinear ABP is one where every input variable labels an edge at most once along any path from s to t.

We consider several interesting sub-classes of syntactically multilinear ABPs. An ABP is said to be *oblivious* if edges between two consecutive layers L_i and L_{i+1} are labelled by a particular variable x_j. We define an important sub-class of syntactically multilinear ABPs, Read-once Oblivious ABPs, as follows:

Definition 3. A read-once oblivious ABP (ROABP) P is an ABP such that there exists a fixed ordering $\pi \in S_n$ in which the input variables are read, along every s to t path. Edges between any two consecutive layers $L_i,\ L_{i+1},\ i \in [0,n]$ are labelled by the variable $x_{\pi(i)}$ or a constant from the field.

An interval on the set $\{1,\ldots,n\}$ with end-points $i,j \in [n]$, can be defined as $I = [i,j],\ i < j$, where $I = \{\ell \mid i,j \in [n], i \le \ell \le j\}$. An *interval of variables* X_{ij} is defined such that $X_{ij} \subseteq \{x_\ell \mid \ell \in I,\ I = [i,j]\}$, where I is an interval on the set $\{1,\ldots,n\}$. For an ordering $\pi \in S_n$, we define a π-interval of variables, $X_{ij} \subseteq$

$\{x_{\pi(i)}, x_{\pi(i+1)}, \ldots, x_{\pi(j)}\}$. An interval ABP, defined by Arvind and Raja (2016), is a syntactically multilinear ABP where each node computes a polynomial defined on an interval of variables. A restricted form of this model was defined by Ramya and Rao (2019a), denoted by the term *strict-interval ABP*, as follows.

Definition 4. (Ramya and Rao (2019a)) A strict interval ABP P is a syntactically multilinear ABP where we have the following:

1. For any pair of nodes u and v in P, the indices of variables occurring in the sub-program $[u, v]_P$ is contained in some π-interval I_{uv} called the *associated interval* of $[u, v]_P$; and
2. for any pairs of sub-programs of the form $[u, v]_P,\ [v, w]_P$, the associatedπ-intervals of variables are disjoint, i.e., $I_{uv} \cap I_{vw} = \emptyset$.

It may be noted that in a strict interval ABP, intervals associated with each sub-program need not be unique. We assume that the intervals associated are largest intervals with respect to set inclusion such that condition 2 in the definition above is satisfied.

2.2 Parameterized Complexity Theory: Basic Definitions

In this section, we go through some basic concepts of parameterized complexity theory, as defined by Downey and Fellows (1999) and Flum and Grohe (2006). Since a parameterized view of algebraic complexity would require us to encounter parameterized versions of the computational problems in algebraic complexity, like Parameterized PIT , we first visit the formal definition of parameterized computational problems.

Definition 5. A *parameterized problem* is a set $P \subseteq \Sigma^* \times \mathbb{N}$, where Σ is a finite alphabet. If $(x, k) \in \Sigma^* \times \mathbb{N}$ is an instance of a parameterized problem, we refer to x as the input and k as the parameter.

The notion of parameterized tractability, which is the aim of parameterized algorithm design, is known as *fixed parameter tractability* (referred to as *FPT* in this thesis). Formally, this notion is defined as follows.

Definition 6. (Flum and Grohe (2006)) A parameterized problem $P \subseteq \Sigma^* \times \mathbb{N}$ is fixed-parameter tractable if there is a computable function $g : \mathbb{N} \to \mathbb{N}$ and an algorithm that, given a pair $(x, k) \in \Sigma^* \times \mathbb{N}$, decides if $(x, k) \in P$ in at most $g(k)\text{poly}(n)$ steps, where n is the length of input.

FPT is the class of parameterized problems that are fixed-parameter tractable and similar to the non-parameterized class P it contains all problems for which an efficient parameterized algorithm is known.

2.3 Parameterized Algebraic Complexity

In this section we define the parameterized view of arithmetic circuits and algebraic complexity theory in general. For this purpose one must consider different possible parameterizations of polynomials or arithmetic circuits computing them. It is possible to study arithmetic circuits with a generic parameter k, as done by Engels in his thesis Engels (2016) and later in Bläser and Engels (2019). In Bläser and Engels (2019), the authors defined parameterized circuit classes with respect to a generic parameter and defined notions of efficiency and reductions between problems in these parameterized algebraic classes, akin to the notions in the boolean setting defined by Downey and Fellows (1999). The authors obtained families of polynomials parameterized by a generic parameter that were complete for the parameterized algebraic classes defined by them.

However, considering arithmetic circuits with a specific parameter that might represent a property of circuits or polynomials might provide more insight about arithmetic computation. In this direction, Müller (2008) considered algebraic computational problems like PIT with an arithmetic circuit given as input, parameterized by number of input variables, number of product gates along any root to leaf path (i.e., multiplicative depth) and degree of the polynomial.

Among these, degree of the polynomial seems interesting because it can be used as a parameter even when the polynomial is given as a black-box. Existing literature in parameterized algorithms for solving graph theoretic problems, as in Koutis (2008), Williams (2009), Fomin *et al.* (2012), also use polynomials parameterized by the degree, by reducing the graph theoretic problem to a computational problem on the polynomial. We need to inspect if degree of the polynomial is suitable for obtaining interesting results in parameterized algebraic complexity theory.

A circuit C of FPT size (size $g(k)n^{O(1)}$) where every gate computes an n-variate polynomial of degree at most k can compute polynomials where the absolute value of the coefficient is as large as $2^{2^{n^{O(1)}}}$ even when the constants from the field, taken as input in the circuit, are limited to the set $\{-1, 0, 1\}$. In this case, the evaluation of the polynomial p cannot be done in FPT-time. A more suitable parameter would be one that takes into account the coefficient size as well. Thus, the *syntactic degree* of a polynomial computed by an arithmetic circuit, also known as the *formal degree* (Kayal *et al.* (2014)), is used as a standard parameter for arithmetic circuits. It is defined as follows:

Definition 7. The syntactic degree of each gate v in an arithmetic circuit C_p computing a polynomial $p \in \mathbb{F}[X]$ is denoted by $syndeg$ such that,

$$syndeg(v) = \begin{cases} 1 \quad \text{if } v \text{ is an input gate} \\ \max\{syndeg(v_1), syntdeg(v_2)\} \quad \text{if } v = v_1 + v_2 \\ syndeg(v_1) + syndeg(v_2) \quad \text{if } v = v_1 \times v_2 \end{cases}$$

where v_i is the polynomial computed at the corresponding gate g_i. The syntactic degree of the polynomial p is the syntactic degree of the output gate of C_p, the circuit computing p.

An arithmetic circuit C is said to be of syntactic degree k if the maximum syntactic degree of any gate in the circuit C is k. As every constant in a circuit has degree 0 but syntactic degree 1, a circuit of syntactic degree d with constants from $\{1, 0, -1\}$ has monomials of degree at most d with the value of coefficients bounded by 2^d. Thus, bounding the syntactic degree of the circuit seems to be a natural restriction on the circuit and ensures FPT-time computation of a polynomial of degree k. Figure 2.3 illustrates an arithmetic circuit with syntactic degree (*syndeg*) values visualised for each gate on a root to leaf path.

Similar to non-parameterized algebraic complexity, parameterized families of polynomials are grouped into parameterized circuit classes, defined as follows.

Definition 8 (Parameterized Circuit classes)**.** A parameterized class of circuits $\mathcal{C}_k$ has

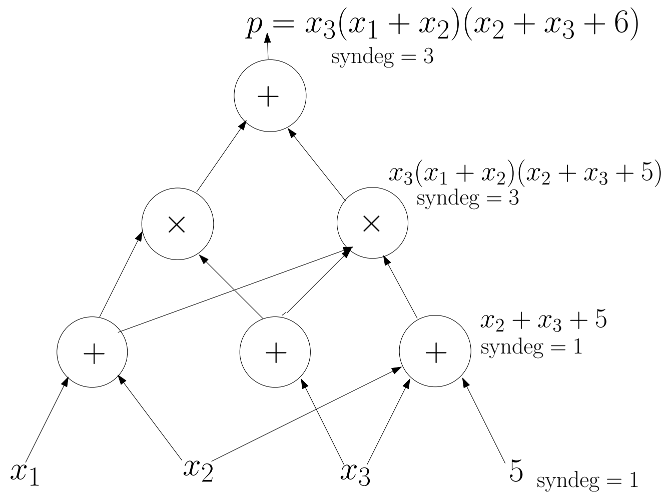


Figure 2.3: Example arithmetic circuit with *syndeg* values for a root to leaf path.

alternate layers of sum and product gates, with the top gate being a sum gate. The fan-in of each product gate in the circuit is bounded by $g(k)$, where $g : \mathbb{N} \to \mathbb{N}$ is any computable function. Here k is the parameter of the circuit or the polynomial computed by the circuit.

One of the parameterized computational problems encountered in this text is Parameterized Arithmetic Circuit Identity Testing, or para-ACIT. ACIT is the problem of testing whether a given arithmetic circuit C computes a polynomial p that is identically zero. There have been several parameterized variants of ACIT studied in the literature Müller (2008). However the problem para-ACIT was first studied by Chauhan and Rao (2015) with the *syntactic degree* of the arithmetic circuit as a parameter.

Having defined *syntactic degree* as the parameter, we now define a degree-parameterized family of polynomials and note its properties.

Definition 9. Let $k = k(n)$. A family $(p_n)_{n\geq 0}$ of polynomials over $\mathbb{F}$ is said to be degree-k parameterized if

- There is a $c > 0$ such that p_n is an n^c variate polynomial for every $n \geq 0$;
- Degree of p_n is bounded by $k = k(n)$ for every $n \geq 0$; and
- The absolute value of the coefficients of p_n is bounded by $2^{g(k)n^c}$, for some function g that depends only on k.

A degree parameterized polynomial family $(p = (p_n)_{n\geq 0}, k)$ with $k = k(n)$ as the parameter is said to be *fixed parameter tractable* if for every $n \geq 0$, there is an arithmetic circuit C_n of syntactic degree at most k and of size $g(k)n^c$ computing p_n where g is a function of k and c is a constant.

Now, we define the problem of parameterized white-box PIT as para-ACIT.

Problem 1. para-ACIT

INPUT: An n-variate, degree-k parameterized polynomial p, given as an arithmetic circuit C.

PARAMETER: k

OUTPUT: YES if $p \equiv 0$.

Chauhan and Rao (2015) have shown the membership of the para-$\overline{\text{ACIT}}$ problem in W[P]-RFPT, a parameterized class of problems with efficient randomised algorithms, analogous to the non-parameterized class RP. In Ghosal *et al.* (2017), we show that when restricted to depth three circuits, this problem is fixed-parameter tractable.

2.4 Lower Bound Techniques and Complexity Measures

Let us recall the arithmetic circuit lower bound problem formally: Given an explicit n-variate polynomial f and a circuit class $\mathcal{C}$ as input, we want to know the $\min_{C \in \mathcal{C}}$ size(C), such that the circuit C computes the explicit polynomial f.

In this section we formally go through the standard method for obtaining lower bound on arithmetic circuits. The following steps will make the process clear.

A polynomial p computed by any circuit C in the given class of circuits $\mathcal{C}$ can be considered as being made of smaller building block polynomials $p_1, p_2, \ldots, p_s$ for some $s > 0$, that are also in the class $\mathcal{C}$, where $p = p_1 + p_2 + \ldots + p_s$ or $p = \prod_{i=1}^{s} p_i$. Here, s also denotes the fan-in of the top gate of the circuit C. If we can show that for a circuit $C \in \mathcal{C}$ to compute f, s must be large, then we have the required lower bound, since the total number of gates in the circuit is larger than s.

For this purpose, it is necessary to define a complexity measure on polynomials (Nisan (1991), Saptharishi *et al.* (2016)), which we denote by the function $\mu : \mathbb{F}[X] \to \mathbb{R}^+$, such that $\mu(p)$ is a positive real value representing the complexity of the polynomial p. As simple polynomials (for example, linear forms) would take low values for the

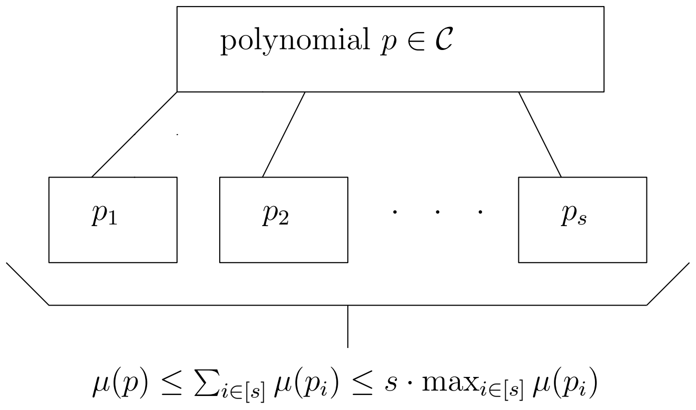

polynomial $p \in \mathcal{C}$
p_1
p_2
$\cdots$
p_s
$\mu(p) \leq \sum_{i\in[s]} \mu(p_i) \leq s \cdot \max_{i\in[s]} \mu(p_i)$

Figure 2.4: Standard lower bound method visualised for $p = p_1 + \ldots + p_s$.

measure, so the measure is easy to calculate for such polynomials. Typically, the well-studied measures considered in the literature have the property that if $p = p_1 + p_2$, then $\mu(p) \leq \mu(p_1) + \mu(p_2)$ and if $p = p_1 \cdot p_2$, then $\mu(p) \leq \mu(p_1) \cdot \mu(p_2)$. These properties are known as *sub-additivity* and *sub-multiplicativity* respectively. Now, in the context of the circuit class $\mathcal{C}$, we consider a polynomial p that can be computed in polynomial size by any arbitrary circuit C in $\mathcal{C}$.

If p is such that $p = p_1 + \ldots + p_s$, then $\mu(p) \leq \mu(p_1) + \ldots + \mu(p_s) \leq s \cdot \mu(p_i)$, where p_i has the largest value of the complexity measure, $\mu(p_i)$. As it is easier to calculate μ for p_i than p, we obtain an upper bound on $\mu(p_i)$ to yield the upper bound of $s \cdot \mu(p_i)$ for any polynomial that can be efficiently computed by a circuit $C \in \mathcal{C}$.

Then, if a large lower bound on $\mu(f)$ is obtained for the explicitly given polynomial f, we have $s \cdot \mu(p_i) \geq \mu(f)$, since f is computed by an arbitrary circuit C in $\mathcal{C}$. From this expression, it is possible to show that, in order to compute f, s needs to be large. This lower bound on s then translates to be the lower bound on the size of the circuit C computing f. Figure 2.4 illustrates the lower bound method for a sub-additive measure μ.

In case the complexity measure μ is not sub-additive and sub-multiplicative, the above standard method may not be useful, and a novel method for obtaining lower bounds will be necessary.

In this thesis, we study multiple complexity measures that are useful for proving lower bounds. One of the earliest such definitions of complexity measures of polynomials is the rank of the partial derivative matrix of a polynomial.

2.4.1 Partial Derivative Matrix of a polynomial

Nisan (1991) defined the partial derivative matrix of a polynomial, considered its rank as a complexity measure for non-commutative polynomials and proved exponential lower bounds for the size of non-commutative formulas and ABPs. Raz (2009) considered a variant of the partial derivative matrix and proved super polynomial size lower bounds for multilinear formulas. We describe the partial derivative matrix introduced by Raz (2009) in more detail.

A partition of X is an injective map $\varphi : X \to Y \cup Z$, where Y and Z are two disjoint sets of variables such that $|X| = |Y \cup Z|$. Let $\varphi : X \to Y \cup Z$ be a partition and $X = X_1 \cup X_2$ where $X_1,\ X_2$ are disjoint sets. Then $\varphi\mid_{X_1}: X_1 \to Y \cup Z$ is defined such that $\forall x \in X_1,\ \varphi\mid_{X_1}(x) = \varphi(x)$.

An *equi-partition* is a partition $\varphi : X \to Y \cup Z$ such that $|Y| = |Z| = |X|/2$. For our convenience, we assume that the number of variables $|X| = n$ is an even number. We use the partial derivative matrix defined by Raz for polynomials parameterized by degree, k in Chapter 5.

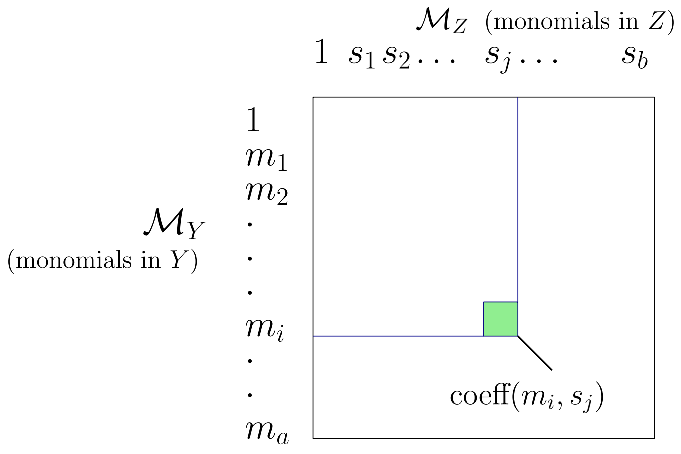
Definition 10 (Raz (2009))**.** Let $f \in \mathbb{F}[x_1, \dots, x_n]$ be a polynomial of degree k, $\varphi : X \to Y \cup Z$ be a partition of the input variables of f. Then the partial derivative matrix of f with respect to φ, denoted by M_{f^φ} is an $A \times B$ matrix, where the rows are indexed by the set of all multilinear monomials μ in the variables Y, and columns indexed by the set of all multilinear monomials ν in the variables in Z, i.e., ., $A = \sum_{i=0}^{k} \binom{|Y|}{i}$ and $B = \sum_{i=0}^{k} \binom{|Z|}{i}$.

For monomials μ and ν respectively in variables Y and Z, the entry $M_{f^\varphi}(\mu, \nu)$ is the coefficient of the monomial $\mu\nu$ in f.

Figure 2.5 is an example of a partial derivative matrix for a polynomial f expressed as $f^\varphi = \sum_{i,j} \mathrm{coeff}(m_i, s_j) m_i s_j$ under the partition $\varphi : X \to Y \cup Z$ where m_i is a monomial in variables mapped to Y, s_j is a monomial in variables mapped to Z.

We modify the definition to include row and column indexing monomials of degree at most k for proving parameterized multilinear lower bounds in Chapter 5.

For a multilinear polynomial $p \in \mathbb{F}[X]$ and an equi-partition φ, let $\mathsf{rank}_\varphi(p)$ be the rank of the matrix M_{p^φ} over $\mathbb{F}$.



$\mathcal{M}_Z$ (monomials in Z)
$1\ s_1 s_2 \ldots\ s_j \ldots\ s_b$
$1\ m_1\ m_2 \cdot \cdot \cdot\ m_i \cdot \cdot\ m_a$
$\mathcal{M}_Y$ (monomials in Y)
coeff(m_i, s_j)

Figure 2.5: Partial derivative matrix for a polynomial $f = \sum_{i,j} \text{coeff}(m_i, s_j) m_i s_j$, $m_i,\ s_j$ are monomials in variables mapped to Y and Z respectively.

The following fundamental properties of the rank of a partial derivative matrix was given by Raz (2009).

Lemma 1. *Let f, g and h be multilinear polynomials of degree at most k in $\mathbb{F}[X]$. Then,*

(Sub-additivity): *If $f = g + h$, then $\forall \varphi : X \to Y \cup Z$, $\mathsf{rank}_\varphi(f) \leq \mathsf{rank}_\varphi(g) + \mathsf{rank}_\varphi(h)$. For $\mathsf{var}(g) \cap \mathsf{var}(h) = \emptyset$, $\mathsf{rank}_\varphi(f) = \mathsf{rank}_\varphi(g) + \mathsf{rank}_\varphi(h)$.*

(Sub-multiplicativity): *If $f = g \times h$, $\forall \varphi : X \to Y \cup Z$, we have $\mathsf{rank}_\varphi(f) \leq \mathsf{rank}_\varphi(g) \times \mathsf{rank}_\varphi(h)$. For $\mathsf{var}(g) \cap \mathsf{var}(h) = \emptyset$, $\mathsf{rank}_\varphi(f) = \mathsf{rank}_\varphi(g) \times \mathsf{rank}_\varphi(h)$.*

Proof. Let $\varphi : X \to Y \cup Z$ be any partition of X , $A = \sum_{i=0}^{k} \binom{|Y|}{i}$ and $B = \sum_{i=0}^{k} \binom{|Z|}{i}$.

Sub-additivity: Suppose that $\overline{M_{g^\varphi}}$ and $\overline{M_{h^\varphi}}$ be $A \times B$ matrices obtained respectively from M_{g^φ} and M_{h^φ} by adding additional zero entries. Then $M_{f^\varphi} = \overline{M_{g^\varphi}} + \overline{M_{h^\varphi}}$ and hence $\mathsf{rank}_\varphi(g + h) \leq \mathsf{rank}_\varphi(g) + \mathsf{rank}_\varphi(h)$ as the rank of a matrix is a sub-additive function.

Additionally, if $\mathsf{var}(g) \cap \mathsf{var}(h) = \emptyset$, then for any two monomials $m_1 \in \mathbb{F}[Y]$ and $m_2 \in \mathbb{F}[Z]$, either $\overline{M_{g^\varphi}}[m_1, m_2] = 0$ or $\overline{M_{h^\varphi}}[m_1, m_2] = 0$. Therefore, $\mathsf{rank}_\varphi(g + h) = \mathsf{rank}_\varphi(g) + \mathsf{rank}_\varphi(h)$.

Sub-multiplicativity: Let g and h be variable disjoint, $\mathsf{var}(g) \cap \mathsf{var}(h) = \emptyset$. Let $\varphi|_g : X_g \to Y_1 \cup Z_1$, and $\varphi|_h : X_h \to Y_2 \cup Z_2$, where $X_g = \mathsf{var}(g)$, $X_h = \mathsf{var}(h)$, $Y = Y_1 \cup Y_2$, $Z = Z_1 \cup Z_2$. Let, $\mathsf{M} = M_{g^{\varphi|_g}} \otimes M_{h^{\varphi|_h}}$ where $\otimes$ denotes the tensor product.

Note that each row index of M can be written as $m_{11}m_{12}$, a product of a multilinear monomial m_{11} in variables in Y_1, and a multilinear monomial m_{12} in variables in Y_2, respectively. Similarly, each column index of M can be written as $m_{21}m_{22}$, a product of a monomial in variables in Z_1, m_{21} and a monomial in variables in Z_2, m_{22} respectively. Now, M_{f^φ} is the sub-matrix of M obtained by removing rows and columns that are indexed by monomials of degree larger than k. Also, rows and columns of M that are indexed by monomials of degree larger than k will have no non-zero entries, for, $f = g \times h$ and f is a degree k multilinear monomial. Hence, we conclude that $\mathsf{rank}_\varphi(f) = \mathsf{rank}_\varphi(g) \cdot \mathsf{rank}_\varphi(h)$. $\square$

Rank upper bound for degree-k polynomials

Lemma 2. *For any equi-partition $\varphi : X \to Y \cup Z$, and any multilinear polynomial p of degree k, we have* $\mathsf{rank}_\varphi(p) \leq (k+2)\binom{n/2}{k/2}$.

Proof. Let $p \in \mathbb{F}[x_1, \ldots, x_n]$ be a degree-k multilinear polynomial. We fix an arbitrary equi-partition $\varphi : X \to Y \cup Z$ with $|Y| = |Z| = n/2$.

For $d \leq k$ let A_d be a matrix constructed from M_{p^φ} such that rows labelled by degree d monomials in the variables $x \in X$ such that $\varphi(x) \in Y$ are copied from M_{p^φ}, and all other rows correspond to all zero entries. We can now express $M_{p^\varphi} = A_0 + A_1 + \cdots + A_k$.

Then by sub-additivity, $\mathsf{rank}_\varphi(p) \leq \sum_{d=0}^{k} \mathsf{rank}(A_d)$. Since p has degree bounded by k, all but $\binom{n/2}{k-d}$ columns of A_d are zero columns. Thus $\mathsf{rank}(A_d) \leq \min\{\binom{n/2}{d}, \binom{n/2}{k-d}\}$. Substituting these values for $\mathsf{rank}_\varphi(p)$, we have, $\mathsf{rank}_\varphi(p) \leq 2\sum_{d=0}^{k/2}\binom{n/2}{d} \leq (k+2)\binom{n/2}{k/2}$.

This proof holds for $k \leq n/2$. Since in the parameterized domain, k is typically much smaller than n, the above calculations are sufficient for us. $\square$

Coefficient Matrix of a polynomial

In Chapter 6, we modify this measure by Raz to include even non-multilinear monomials as row and column indices of degree at most the degree of the polynomial. We call this measure the *coefficient matrix* of a polynomial.

Definition 11. Let $f \in \mathbb{F}[x_1, \ldots, x_n]$ be a polynomial of degree d, $\varphi : X \to Y \cup Z$ be a partition of the input variables of f. Then the coefficient matrix M_{f^φ} has its rows indexed by monomials μ of degree at most d in variables in Y, and columns indexed by monomials ν of degree at most d in variables in Z. For monomials μ and ν respectively in variables Y and Z, the entry $M_{f^\varphi}(\mu, \nu)$ is the coefficient of the monomial $\mu\nu$ in f^φ.

The fundamental properties of sub-additivity and sub-multiplicativity of the rank of the coefficient matrix follow directly from the related definition of the Partial Derivative matrix Raz (2009).

2.4.2 Dimension of Partial Derivatives

The partial derivative matrix considers all possible partial derivatives of the polynomial. A measure more suitable to polynomials of the form $p = Q_1^k + \ldots + Q_s^k$, where all $Q_i,\ i \in [s]$ are low degree polynomials, k is the degree of p and p is computable by a depth-4 circuit is the dimension of the space spanned by partial derivatives of p, since this measure increases with the degree of the $Q_i,\ i \in [s]$ polynomials.

Definition 12. For a polynomial $f \in \mathbb{F}[X]$, let $\partial^{=m}(f)$ be the set of all m^{th} order derivatives of f. The vector space spanned by the polynomials in the set $\{\partial^{=m}(f)\}$ is the space of partial derivatives of f. The dimension of this space is the dimension of partial derivatives of f.

This measure is sub-additive and sub-multiplicative since the operation of taking partial derivatives of a fixed order of a polynomial f has the sub-additivity and sub-multiplicativity properties. We use this measure in Chapter 4 for proving a lower bound against depth-5 powering circuits with small top product gate fan-in.

Now that we have gone through the standard approach for proving lower bounds and the complexity measures used in this thesis, we proceed to inspect the hard polynomials that are directly used or adapted to construct a different polynomial.

2.5 Some Explicit Polynomials

Polynomials that exhibit full rank of partial derivative matrix under all or a huge fraction of partition functions on input variables can be thought of as *high complexity* or *hard* polynomials. In this section we visit some of the well-known hard polynomials in the literature which we have also used in obtaining our lower bound results.

Raz and Yehudayoff (2008) defined a multilinear hard polynomial and show a super-polynomial lower bound against syntactically multilinear formulas computing it. To define this polynomial we denote an interval $\{a \mid i \leq a \leq j, a \in \mathbb{N}\},\ i,j \in \mathbb{N}$ by $[i,j]$, and consider the sets of variables $X = \{x_1, \ldots, x_{2n}\}$, $W = \{w_{i,\ell,j}\}_{i,\ell,j\in[2n]}$. We denote it as the Raz-Yehudayoff polynomial and define it as follows.

Definition 13 (**Raz-Yehudayoff polynomial**, Raz and Yehudayoff (2008))**.** Let us consider $f_{ij} \in \mathbb{F}[X, W]$ defined over the interval $[i,j]$. If the length of the interval $[i,j]$ is such that $|[i,j]| = 1$, $f_{ij} = 0$. For $|[i,j]| > 0$,

$$f_{ij} = (1 + x_i x_j) f_{i+1,j-1} + \sum_{\ell\in[i+1,j-2]} w_{i,\ell,j} f_{i,\ell} f_{\ell+1,j},$$

where, for all ℓ, lengths of $[i,\ell],\ [\ell+1,j]$ are even and smaller than $[i,j]$. For all intervals $[i,j]$ such that $i > j$, we define $f_{ij} = 0$.

The polynomial $f_{1,2n}$ is defined as the Raz-Yehudayoff polynomial f_{RY}.

Raz and Yehudayoff showed the polynomial f_{RY} to be attaining full rank under a random partition over the extension field where the variables in W are formal variables. We formally define their statement as follows.

Proposition 1. (Raz and Yehudayoff (2008)) Let $\mathbb{G} = \mathbb{F}(W)$ be the field of rational functions over the field $\mathbb{F}$ and the set of variables W. Then the polynomial $f_{\mathsf{RY}} \in \mathbb{G}[X]$ attains full-rank of the partial derivative matrix, against any partition $\varphi \sim \mathcal{D}$, where $\mathcal{D}$ is the distribution over all equi-partitions $\varphi : X \to Y \cup Z$.

Later, Dvir *et al.* (2012) defined a polynomial that is hard i.e., full rank with respect to a special class of partition functions, denoted by the term *arc-partitions*.

Definition 14. (Arcs) Let us identify the set of variables $X = \{x_0, \ldots, x_{n-1}\}$ with the set $V = \{0, \ldots, n-1\}$ of indices. Let us consider the n-cycle graph $G = (V, E)$ where each edge $e \in E,\ e = (i, (i+1) \mod n), \forall i \in [n]$. Then $\forall i \neq j$ in the n-cycle we denote $[i, j]$ to be the arc from i to j.

Given the above definition of arcs, we can proceed to formally define the construction of an arc-partition by randomly sampling $n/2$ disjoint pairs according to a distribution of pairs such that the elements in each pair lie in different partitions of the input variables.

Definition 15. (Arc partitions) Let us consider a random pairing constructed in $n/2$ steps in the following manner: Assuming a pairing $(P_1, \ldots, P_t)$ constructed in $t < n/2$ steps, where $P_1 = (0, 1)$, $[L_t, R_t]$ is the interval spanned by $\cup_{i \in [t]} P_i$ and the random pair P_{t+1} is constructed such that

$$P_{t+1} = \begin{cases} (L_t - 2, L_t - 1) \text{ with probability } 1/3, \\ (L_t - 1, R_t + 1) \text{ with probability } 1/3, \\ (R_t + 1, R_t + 2) \text{ with probability } 1/3, \end{cases}$$

and therefore, $[L_{t+1}, R_{t+1}] = [L_t, R_t] \cup P_{t+1}$. Then a pairing $(P_1, \ldots, P_{n/2})$ from the distribution of pairings $\mathcal{D}$ constitutes an arc partition Π where for each $P_i = (x_j, x_k)$, either $\Pi(x_j) \in Y,\ \Pi(x_k) \in Z$ or $\Pi(x_j) \in Z,\ \Pi(x_k) \in Y$ with equal probability.

The uniform distribution on all arc-partitions defined from as above is denoted by $\mathcal{D}$. Given the above setting, the following definition of the hard polynomial can be seen to be considering all pairings along the n-cycle.

Definition 16. (**DMPY polynomial**, Dvir *et al.* (2012)) The polynomial $\widehat{f}$, given an ABP F computing it, is defined as follows:

$$\widehat{f} = \sum_{p \in \mathcal{P}} \prod_{(u,v)=e \in p} \lambda_e (x_u + x_v),$$

where $\mathcal{P}$ is the set of all paths from source s to terminal t in F and $\lambda_e \in \mathbb{F}$ are constants that take value 0 or 1.

The authors show that for any arc-partition Π sampled from $\mathcal{D}$ uniformly at random, there is a path p in F such that the polynomial $\widehat{f_p} = \prod_{(u,v)=e \in p} \lambda_e (x_u + x_v)$. The polynomial $\widehat{f_p}$ where p is the path where the edges correspond to pairs in the arc-partition

Π, will have full arc-rank. As $\hat{f} = \sum_{p \in \mathcal{P}} \hat{f}_p$, therefore, by sub-additivity of rank and because $\prod_{e \in p} \lambda_e$ is distinct for every path p, f is also of full arc-rank. We can set $\lambda_e = 1$ for all edges $e \in p$, where p corresponds to Π, λ_e is set to zero for all other edges. Let T be the set of all formal variables $\lambda_e,\ e \in F$. We formally state the full arc-rank property as follows.

Proposition 2. (Dvir *et al.* (2012)) The DMPY polynomial $\hat{f} \in \mathbb{G}[X]$ has full arc-rank over the rational function field $\mathbb{G} = \mathbb{F}[T]$ under any arc-partition Π sampled uniformly at random from the uniform distribution on all arc-partitions, $\mathcal{D}$.

CHAPTER 3

ON BOUNDED-READ MODELS OF COMPUTATION

3.1 Introduction

In this chapter, we consider the well-studied restriction of multilinear models by fixing the order or the number of times input variables are read. The earliest study of restricted-read multilinear models follows from Nisan (1991), who obtained an exponential lower bound on non-commutative ABPs computing the palindrome polynomial. However, since ROABPs read every input variable only once along every computational path, the computation in a ROABP does not make use of commutativity of the product of variables. Thus, Nisan's lower bound also holds for this model. Later, Kayal *et al.* (2016) obtained a separation between the classes of ROABPs and depth-3 multilinear circuits. Ramya and Rao (2018) obtain an exponential lower bound against the size of the sum of ROABPs computing the polynomial defined by Raz and Yehudayoff Raz and Yehudayoff (2008). The Raz-Yehudayoff polynomial attains full value, which is exponential in n, of the rank of the partial derivative matrix against all possible partitions of the input variables. The explicitly defined hard polynomial by Dvir *et al.* (2012), on the other hand, attains the full value of rank against a distribution of partition functions of a restricted nature, known as *arc partitions*. Hence, we are motivated to consider if the model of sum of ROABPs has the power to efficiently compute the DMPY polynomial.

Prior work on multilinear formulas have been majorly focused on the class of syntactically multilinear formulas, as seen in the articles Raz (2006), Raz (2009). In Ramya and Rao (2019*b*), the authors proved an exponential lower bound against the size of the sum of read-once formulas computing the Raz-Yehudayoff polynomial. Read-k formulas are a generalization of the read-once formula model, and it is interesting to study the difference in computational power between the classes of sum of ROFs and read-k formulas. We prove an exponential lower bound against the sum of ROFs computing a hard polynomial that can be efficiently computed by ROABPs and read-k formulas where k is bounded.

Interval ABPs were defined and studied in Arvind and Raja (2016), who proved an exponential lower bound against the model, assuming the sum of squares conjecture. In Ramya and Rao (2019*a*), the authors defined the more restricted model of strict-interval ABPs and proved an exponential lower bound against the model computing the Raz-Yehudayoff polynomial. It is therefore, a natural question to study whether the model of strict-interval ABPs has the same computational power as ROABPs. We show that, in fact, strict-interval ABPs denote the same class of circuits as ROABPs.

3.2 Chapter Outline

In Section 3.3, we prove a sub-exponential lower bound on the size of sum of ROABPs computing the DMPY polynomial.

Theorem 1. *Let $f = f_1 + \ldots + f_s$, each $f_i \in \mathbb{F}[X]$ being computable by a ROABP of size poly(n). If f is the full arc-rank polynomial $\widehat{f}$ defined by Dvir* et al. *(2012) then $s = exp(\Omega(n^\epsilon))$ for $\epsilon > 1/500$.*

Next, we obtain a depth reduction for the model of interval formulas in Section 3.4.

Theorem 4. *Let $f \in \mathbb{F}[X]$ be a polynomial computed by an interval formula F of size s and depth d. Then f can also be computed by an interval formula of size poly(s) and depth $O(\log s)$.*

In Section 3.5 we show that the class of strict interval ABPs are the same as the class of ROABPs.

Theorem 5. *The class of Strict Interval ABPs is equivalent to the class of ROABPs.*

The construction follows from Lemma 6.

Finally, we prove an exponential lower bound on the sum of ROFs computing an explicit polynomial computable by a polynomial size ROABP in Section 3.6.

Theorem 6. *Let $f_1, \ldots, f_s$ be read-once polynomials such that $f_{PRY} = f_1 + f_2 + \cdots + f_s$, then $s = 2^{\Omega(n)}$, where the polynomial f_{PRY} is defined as in Equation 3.1.*

3.3 Lower Bound on Sum of ROABPs computing DMPY polynomial

In this section we prove a sub-exponential lower bound against the size of sum of read-once oblivious ABPs computing the hard polynomial constructed in Dvir *et al.* (2012). This shows a separation between syntactically multilinear ABPs and sum of ROABPs.

We recall an *arc partition* Π to be a partition function, $\Pi : X \to Y \cup Z$, based on a pairing $(P_1, \ldots, P_{n/2})$ of the input variables such that, for a pair $P = (x_i, x_j)$, $\varphi(x_i) \in Y$, $\varphi(x_j) \in Z$, or vice versa. Our aim is to give an upper bound on the maximum rank of ROABPs under an arc partition. We prove the following theorem in this section:

Theorem 1. *Let $f = f_1 + \ldots + f_s$, each $f_i \in \mathbb{F}[X]$ being computable by a ROABP of size poly(n). If f is the full arc-rank polynomial $\widehat{f}$ defined by Dvir* et al. *(2012) then $s = exp(\Omega(n^\epsilon))$ for $\epsilon > 1/500$.*

We refer to the rank of the coefficient matrix of the sum of ROABPs against an arc-partition as the *arc-rank*. We analyse the arc-rank of the sum of ROABPs against an arc-partition to give a lower bound on the size of the sum necessary to compute $\widehat{f}$.

Let us assume n is even. In order to prove the lower bound, we need to estimate an upper bound on the arc-rank computed by a ROABP. We define the notion of F-arc-partition, F being a ROABP, as follows:

Definition 17. Let us consider an arc partition Q constructed from a ROABP F in the following manner: Let the order of variables appearing in the ROABP be $x_{\sigma(1)}, x_{\sigma(2)}, \ldots, x_{\sigma(n)}$, where $\sigma \in S_n$ is a permutation on n indices. Then, $Q = \{(x_{\sigma(i)}, x_{\sigma(i+1)}) \mid i \in [n],\ i \text{ is odd}\}$ is a F-arc-partition.

We assume $2K \mid n$. Let $S_1, \ldots, S_K$ be a K-coloring of the variable set X, where $x_1, \ldots, x_n$ are ordered according to the ROABP. For every $i \in [K]$, S_i contains the variables $x_{(i-1)n/K+1}, \ldots, x_{in/K}$ according to the ordering on the ROABP. Then $S_1, \ldots, S_K$ is a K-partitioning of the pairs in the F-arc-partition Q. So pairs in Q are monochromatic, whereas the pairs $(P_1, \ldots, P_{n/2})$ on which a random arc-partition Π sampled from $\mathcal{D}$ is based, might cross between two colors.

Our analysis for the ROABP arc-rank upper bound follows along the lines of the analysis for the arc-rank upper bound given by Dvir *et al.* (2012) for syntactic multi-linear formulas. For this analysis we define the set of violating pairs for each color c, $V_c(\Pi)$, that is defined as:

$$V_c(\Pi) = \{\Pi_t \mid |\Pi_t \cup S_c| = 1,\ t \in [n/2]\},$$

where $\Pi_1, \ldots, \Pi_{n/2}$ are pairs in Π. The quantity $G(\Pi) = |\{c \mid |V_c(\Pi)| \geq n^{\frac{1}{1000}}\}|$, representing the number of colors with many violations, is similarly defined. We use the following lemma directly from Dvir *et al.* (2012):

Lemma 3. *Let $K \leq n^{\frac{1}{100}}$, Π be the sampled arc-partition, and $G(\Pi)$ be as defined above. Then, we have,*

$$\mathsf{Pr}_{\Pi \in \mathcal{D}}[G(\Pi) \leq K/1000] \leq n^{-\Omega(K)}.$$

The following measure is used to calculate the arc-rank upper bound for ROABPs.

Definition 18. (Similarity function) Let φ be a distribution on functions $\mathcal{S} \times \mathcal{S} \to \mathbb{N}$, such that $\mathcal{S}$ is the support of the distribution on arc-partitions, $\mathcal{D}$. Let P, Q be arc-partitions sampled independently from $\mathcal{D}$. Then, $\varphi(Q, P) : \mathcal{S} \times \mathcal{S} \to \mathbb{N}$ is the total number of common pairs between two arc-partitions Q and P.

We assume Q to be the F-arc-partition for the ROABP F. For a pair that is not common between Π and Q, we show both the variables in the pair is in the same partition, Y or Z with high probability.

Theorem 2. *Under an arc-partition Π sampled from $\mathcal{D}$ uniformly at random, if $p \in \mathbb{F}[X]$ is the polynomial computed by a ROABP P, then, for the similarity function φ and $\delta > 0$,*

$$\mathsf{Pr}_{\Pi \sim \mathcal{D}}[\varphi(\Pi, Q) \geq n/2 - n^{\delta}] \leq 2^{-o(n)}.$$

Proof Outline: Our argument is the same as Dvir *et al.* (2012). It is being included here for completeness for the parameters here being somewhat different than Dvir *et al.* (2012).

In order to analyse the number of common pairs counted by φ, we consider the K-coloring of F and show that under a random arc-partition Π, the number of crossing pairs are large in number using Lemma 3. Then, we show, this results in large number of pairs having both elements in Y. In order to identify the colors with the high number of crossing pairs, a graphical representation of the color sets is used.

Proof. Dvir *et al.* (2012) construct the graph $H(\Pi)$, where each vertex is a color c such that $|V_c(\Pi)| \geq n^{\frac{1}{1000}}$, and vertices c and d have an edge connecting them if and only if $|V_c(\Pi) \cap V_d(\Pi)| \geq n^{\frac{1}{1500}}$. We know for any two colors $c, d \in [K]$, $|V_c(\Pi) \cap V_d(\Pi)| \leq n^{\frac{1}{1000}}$. So, by definition of $H(\Pi)$, the least degree of a vertex in $H(\Pi)$ is 1. Using this, Dvir *et al.* (2012) prove the following claim:

Claim 1. Let the size of the vertex set of $H(\Pi)$, $V(H(\Pi))$, be M. For any subset U of $V(H(\Pi))$ of size N, $N \geq M/2 - 1$, there is some color $h_{j+1}, j \in [N-1]$ such that in the graph induced on all vertices except $\{h_1, \ldots, h_j\}$, the degree of h_{j+1} is at least 1.

By Claim 1, we have $U \subseteq V(H(\Pi))$, $U = \{c_1, \ldots, c_{M/2-1}\}$ such that this is the set of colours having high number of crossing pairs common with colors not in U. Considering the colors sequentially, given Π, we first examine the pairs crossing from color c_1 to other colors, then c_2 and so on. Therefore, to examine the event E_i for color c_i, we have to estimate $\mathsf{Pr}_{\Pi \sim \mathcal{D}}[E_i \mid E_1, \ldots, E_{i-1}, \Pi]$.

Here, E_i is the event $|Y_{c_i} - |S_{c_i}|/2| \leq n^{\frac{1}{5000}}$, equivalently expressed as $|S_{c_i}|/2 - n^{\frac{1}{5000}} \leq Y_{c_i} \leq |S_{c_i}|/2 - n^{\frac{1}{5000}}$. But for an upper bound, it suffices to analyse the $n^{\frac{1}{1500}}$ crossing pairs from S_{c_i} to S_{c_j} instead of considering the entire set. Let the subset of Y_{c_i} constituted by one end of crossing pairs going to color c_j be P_{ij}. Each element x in a crossing pair $P_t = (x, w)$ is a binomial random variable in a universe of size $\geq n^{\frac{1}{1500}=s}$ with probability $1/2$ of being allotted to the subset Y of the universe. This event is independent of how the c_i colored element of other crossing pairs $P_{t'}$ are allotted. So, $|B_{ij}| = b_j$ is a hypergeometric random variable where B_{ij} contains all such $x \in Y$. By the properties of a hypergeometric distribution, $\mathsf{Pr}_{b_j}[b_j = a] = O(s^{\frac{-1}{2}}) = O(n^{\frac{-1}{3000}})$, where a is a specific value taken by the size of B_{ij}.

Applying the union bound over all colors c_j for the crossing pairs, and taking $b = \sum_{j \in U \setminus \{i\}} b_j$, we have:

$$\mathsf{Pr}_b[s/2 - n^{\frac{1}{5000}} \leq b \leq |S_{c_i}|/2 - n^{\frac{1}{5000}}] \leq 2n^{\frac{1}{5000}} O(n^{\frac{-1}{3000}}) = n^{-\Omega(1)}.$$

Therefore, $\mathsf{Pr}_{\Pi\sim\mathcal{D}}[E_i \mid E_1,\ldots,E_{i-1},\Pi] = n^{-\Omega(\delta)}$.

We want an upper bound for $\mathsf{Pr}[|Y_c - |S_c|/2| \leq n^{\frac{1}{5000}}\forall c \in [K]]$. We have calculated an upper bound for the colors in $[K]$ that were highly connected to each other in $H(\Pi)$. So, we can now estimate the total probability as follows:

$$\begin{aligned}
&\mathsf{Pr}[|Y_c - |S_c|/2| \leq n^{\frac{1}{5000}}\forall c \in [K]]\\
&= \mathsf{E}[n^{-\Omega(G(P))} \mid G(P) > K/1000] + \mathsf{E}[n^{-\Omega(G(P))} \mid G(P) \leq K/1000]\\
&= \mathsf{E}[n^{-\Omega(G(P))} \mid G(P) > K/1000] + n^{-\Omega(K)} \text{ by Lemma 3}\\
&\leq n^{-\Omega(K)}.
\end{aligned}$$

If we consider $\delta = 1/5000$, then:

$$\mathsf{Pr}_{\Pi\sim\mathcal{D}}[\varphi(\Pi,Q) \geq n/2 - n^{\delta}] \leq \mathsf{Pr}[|Y_c - |S_c|/2| \leq n^{\frac{1}{5000}}\forall c \in [K]] \leq n^{-\Omega(K)}$$

Now, in Lemma 3, $K \leq n^{\frac{1}{1000}}$.

Hence, $\mathsf{Pr}_{\Pi\sim\mathcal{D}}[\mathsf{rank}_{\varphi}(M(p_{\Pi})) \geq 2^{n/2-n^{\delta}}] \leq 2^{-cn^{\frac{1}{1000}}\log n} = 2^{-o(n)}$. □

Now, using the above Theorem 2, we can prove the lower bound on the size of the sum of ROABP, s.

Proof. (of Theorem 1) Since the polynomial f is such that each multiplicand is of the form $\lambda_e(x_u + x_v)$, if x_u, x_v are both mapped to the same partition Y or Z, it will reduce the rank of the partial derivative matrix by half. Hence, we have the following:

$$\mathsf{Pr}_{\Pi\sim\mathcal{D}}[\mathsf{rank}_{\varphi}(M(f_{\Pi})) \geq 2^{n/2-n^{\delta}}] = \mathsf{Pr}_{\Pi\sim\mathcal{D}}[\varphi(\Pi,Q) \geq n/2 - n^{\delta}],$$

for some suitable $\delta > 0$.

$$\begin{aligned}1 = \mathsf{Pr}[\mathsf{rank}(M(f_\Pi)) = 2^{n/2}] &\le \mathsf{Pr}[\exists i \in [s],\ \mathsf{rank}(M((f_i)_\Pi)) \ge 2^{n/2}/s] \\ &\le \sum_{i=1}^{s} \mathsf{Pr}[\mathsf{rank}(M((f_i)_\Pi)) \ge 2^{n/2}/s] \\ &\le \sum_{i=1}^{s} \mathsf{Pr}[\mathsf{rank}(M((f_i)_\Pi)) \ge 2^{n/2-n^\delta}] \text{ for some } \delta > 0 \\ &\le s \cdot n^{-\Omega(n^{\frac{1}{1000}})} \\ \implies s &= 2^{\Omega(n^{\frac{1}{1000}} \log n)} = 2^{\Omega(n^{\frac{1}{500}})}.\end{aligned}$$

Therefore we have a lower bound of $2^{\Omega(n^\epsilon)}$ for $\epsilon \ge 1/500$ on the size of sum of ROABPs s. □

3.4 Interval Formulas

We introduce interval formulas as a generalization of read-once formulas. An interval on variable indices, $[i, j],\ i < j$, is an interval corresponding to the set of variables $X_{ij} \subseteq X = \{x_1, \dots, x_n\}$, where $X_{ij} = \{x_p \mid x_p \in X,\ i \le p \le j\}$. Polynomials are said to be defined on the interval $[i, j]$ when the input variables are from the set X_{ij}. When there is no ambiguity, we refer to X_{ij} as an interval of variables $[i, j]$.

This notion of intervals of variables can be extended to permutations on the set of variables X. For example, let π be a permutation on n elements, then a π-interval $[i, j],\ i < j$ is the set of variables $\{x_{\pi(i)}, x_{\pi(i)+1}, \dots, x_{\pi(j)}\}$.

Gates in a read-once formula F can also be viewed as reading an interval of variables according to an order π on the variables i.e., there is a permutation $\pi \in S_n$ such that every gate v in F is a sub-formula computing a polynomial on a π-interval of variables. Thus, interval formulas are a different generalization of read-once formulas where every gate v in the formula F reads an interval of variables in a fixed order. We formally define interval formulas as follows:

Definition 19. (Interval Formulas) An arithmetic formula F is an *interval formula* if for every gate g in F, there is an interval $[i, j],\ i < j$ such that g computes a polynomial in X_{ij} and for every product gate $g = h_1 \times h_2$, the intervals corresponding to h_1 and h_2

must be non-overlapping.

Thus, if a product gate g in F defined on an interval $I = [i, j]$ takes inputs from gates $g_1, \ldots, g_t$, then the gates $g_1, \ldots, g_t$ compute polynomials on disjoint intervals of variables $X_1, \ldots, X_t$ corresponding to intervals $[i, j_1], [j_2, j_3], \ldots, [j_{2t-2}, j]$ respectively, where $\forall p,\ j_p < j_{p+1}$ and $i \leq j_p \leq j$. If g_1, g_2, defined on intervals $I_1,\ I_2$ are input gates to a sum gate g', then the interval I associated with g' is $I = I_1 \cup I_2$. This implies that every gate g in F has an interval $I_g \subseteq [1, n]$ associated with it.

The class of ROFs is weaker than the class of interval formulas, since the polynomial $x_1x_2 + x_2x_3 + x_1x_3$ cannot be computed by ROFs but has a small interval formula computing it. In fact, interval formulas are universal, since any sum of monomials can be represented by an interval formula.

It is not known if every ROF can be converted to a ROF of logarithmic depth. However, we argue, in the following section, that interval formulas can be depth-reduced efficiently.

3.4.1 Depth Reduction

We have the following depth reduction result for general arithmetic formulas given by Brent (1974), who showed that depth of any arithmetic formula can be reduced by allowing its size to be increased polynomially.

Theorem 3. *Brent (1974) Any polynomial p computed by an arithmetic formula of size s and depth d, can also be computed by a formula of size poly(s) and depth $O(\log s)$.*

We know this reduction preserves multilinearity. However, we don't know if Theorem 3 can be modified to preserve the read-k property. We show that the depth reduction algorithm given by Theorem 3 preserves the interval property.

Theorem 4. *Let $f \in \mathbb{F}[X]$ be a polynomial computed by an interval formula F of size s and depth d. Then f can also be computed by an interval formula of size poly(s) and depth $O(\log s)$.*

Proof. We know the underlying structure of any arithmetic formula is a tree. The proof by Brent crucially uses the *tree-separator lemma* Chung (1989). This lemma guarantees that in any arithmetic formula Φ of size s, along any root to leaf path, there is a node g such that the sub-formula Φ_g, rooted at the node g, has size s' such that $s/3 \leq s' \leq 2s/3$. This node g is the *tree-separator node* of Φ.

The construction by Brent (1974) proceeds as follows. We replace the gate g by a new formal variable y. Let the resulting polynomial computed by Φ be $f'(x_1, \ldots, x_n, y)$, where $f(x_1, \ldots, x_n) = f'(x_1, \ldots, x_n, g)$ under the new substitution $y = g$.

As f' is multilinear in y, we can express

$$f'(x_1, \ldots, x_n, y) = y f_1(x_1, \ldots, x_n) + f_0(x_1, \ldots, x_n),$$

where $f_0 = f' \mid_{y=0}$ and $f_1 = f' \mid_{y=1} - f' \mid_{y=0}$. By definition, f_0, f_1 can be computed by multilinear formulas of size $\leq 2s/3$. Now, recursively obtaining small-depth formulas for f_1, f_0, we obtain a $O(\log s)$ depth formula computing f.

However, the above construction does not necessarily preserve the interval property, since the intervals of variables on which f_1 and g are defined, can be overlapping. We overcome this problem by expressing f_1 as products of polynomials over disjoint intervals, each of the intervals being disjoint to the interval corresponding to g.

Construction of a depth-reduced interval formula F' from F:

We assume, without loss of generality, that the interval formula F corresponds to the interval $[1, n]$. As the polynomial computed by F is $f_0 + f_1 \cdot g$, the intervals corresponding to f_1 and g must be disjoint. Let the interval corresponding to g be $I_g = [i, j],\ i < j$. Now, by definition of f_1 and f_0, they are defined on the same interval of variables. We consider the intervals I_0, I_1 such that $I_0 \cup I_1 = [1, n] \setminus [i, j]$, $I_0 = [j + 1, n]$ and $I_1 = [1, i - 1]$. We express $f_1 = f_{1,1} \cdot f_{1,0}$ where $f_{1,1}$ is a polynomial on the interval I_1 and $f_{1,0}$ is a polynomial on the interval I_0.

We consider the root to g (tree-separator node) path ρ in the original formula F containing the node g. All the paths ρ' that join ρ at a sum gate compute polynomials that do not get multiplied by g i.e., these polynomials contribute towards the computation of f_0 and not f_1. For f_1, we will analyse only the paths meeting ρ at product gates. Let us consider a product gate on ρ computing $h_1 \times h_2$, such that h_2 lies on ρ. Since I_g is contained in the interval corresponding to h_2, we claim that the interval corresponding to h_1, I_{h_1} must be either fully contained in I_1 or I_0. This is because, by definition of $I_0,\ I_1$, if I_{h_1} contains variables from both I_0 and I_1, then since $I_g = [i, j]$ where i is greater than any element in I_0 and j is smaller than any element in I_1, $I_g \subseteq I_{h_1}$, which is not possible.

Constructing an interval formula for f_1: We ignore all sum gates on ρ computing $p_1 + p_2$, with p_2 on ρ, by substituting p_1 to zero. The resulting formula is F'. For all product gates on the root to g path ρ, computing $h_1 \times h_2$, where h_2 is on ρ, if $I_{h_1} \subset I_0$, we substitute each h_1 by 1. We also substitute g by 1. The remaining formula F'_1 computes a polynomial $f^{(1)}$ on the interval I_1.

For all product gates of the form $h_1 \times h_2$ where h_2 is on ρ and $I_{h_1} \subset I_1$, we repeat this process by substituting h_1s and g by 1. This remaining interval formula F'_2 computes a polynomial $f^{(2)}$ on interval I_0.

The polynomials $f^{(1)},\ f^{(2)}$ are on the disjoint intervals $I_0,\ I_1$ respectively. By the definition of a product gate in an interval formula, the product gate computing $f_1 \cdot g$ can thus also be seen as computing $f^{(2)} \cdot g \cdot f^{(1)}$. The size of an interval formula computing f_1 is thus size(F'_1) + size(F'_2), which is at most size(F) − size(F_g).

Constructing an interval formula for f_0: We ignore all product gates on ρ computing $h_1 \times h_2$, with h_2 on ρ, by substituting h_1 by 1. We also substitute g by 0. The resulting formula is $\hat{F}$, computing f_0, of size at most size(F) − size(F_g).

Hence, we obtain $f = f^{(1)} \cdot g \cdot f^{(2)} + f_0$.

Analysing size of F': We know $s/3 \leq$ size$(F_g) \leq 2s/3$. Since size$(F_{f_1}) \leq$ size$(F)-$ size(F_g), $s/3 \leq$ size$(F_{f_1}) \leq 2s/3$, where F_{f_1} is the interval formula computing f_1. The same holds for F_{f_0}. Thus, the recursive relation for size is as follows: $\mathsf{Size}(s) \leq 3\mathsf{Size}(2s/3) + 2$, which will yield size$(F') \leq$ poly(s).

The recursive relation for calculating depth is as follows: $\mathsf{Depth}(F) \leq \mathsf{Depth}(F_g) + 2 \implies \mathsf{Depth}(s) = \mathsf{Depth}(2s/3) + 2$, which yields a total depth of $O(\log s)$ for F. □

The notion of intervals of variables corresponding to every sub-formula can be extended to ABPs in the form of Strict-Interval ABPs, where every sub-program corresponds to an interval. In a ROABP P that reads variables in the order $x_1, \ldots, x_n$, any sub-program of P computes a polynomial in the variables $\{x_i, x_{i+1}, \ldots, x_j\}$ for some $i \leq j$. Thus, we can associate an interval $[i, j],\ i \leq j$ of $[1, n]$ with every sub-program of P. This property can be generalized to a multilinear ABP where different paths are allowed to read the variables in different orders, but variables read in every sub-program of the ABP can be associated with an interval. We define such a multilinear ABP as a strict-interval ABP, and thus the class of strict-interval ABPs can be perceived as a generalization of the class of ROABPs.

In the following section, we note that the interval property limits the power of the multilinear ABP, making it same as the class of ROABPs.

3.5 Strict-Interval ABPs

A strict-interval ABP, defined in Ramya and Rao (2019*a*) (See Definition 4), is a restriction of the notion of interval ABPs introduced by Arvind and Raja (2016). In the original definition given by Ramya and Rao (2019*a*), every sub-program in a strict-interval ABP P is defined on a π-interval of variables for some order π, however, without loss of generality, we assume π to be the identity permutation on n variables. Therefore, an interval of variables $[i, j],\ i < j$ here is the set $\{x_i, \ldots, x_j\}$. In this section we show that strict-interval ABPs are equivalent to ROABPs upto a polynomial blow-up in size.

Theorem 5. *The class of strict-interval ABPs is equivalent to the class of ROABPs.*

The proof of Theorem 5 involves a crucial observation that in a strict-interval ABP, variables are read in at most two orders and the nodes that correspond to paths that read in different orders can be isolated. We start with some observations on intervals in $[1, n]$ and the intervals involved in a strict interval ABP.

Let P be a strict-interval ABP over the variables $X = \{x_1, \ldots, x_n\}$. For any two nodes u and v in P, let $I_{u,v}$ be the interval of variables associated with the sub-program of P with u as the start node and v as the terminal node. For two intervals $I = [a, b], J = [c, d]$ in $[1, n]$, we say $I \preceq J$, if $b \leq c$. Note that any two intervals I and J in $[1, n]$ are comparable under $\preceq$ if and only if either they are disjoint or the largest element in one of the intervals is hte smallest element in the other. This defines a natural transitive relation on the set of all intervals in $[1, n]$. The following is a useful property of $\preceq$:

Observation 1. Let I, J and J' be intervals over $[1, n]$ such that $I \preceq J$ and $J' \subseteq J$. Then $I \preceq J'$.

Proof. Let $I = [a, b], J = [c, d]$ and $J' = [c', d']$. As $I \preceq J$, we have $b \leq c$. Further, since $J' \subseteq J$, we have $c \leq c'$ and $d' \leq d$. Therefore, $b \leq c'$ and hence $I \preceq J'$. $\square$

We begin with an observation on the structure of intervals of the sub-programs of P. Let v be a node in P. We say v is an *ascending* node, if $I_{s,v} \preceq I_{v,t}$ and a *descending* node if $I_{v,t} \preceq I_{s,v}$.

Observation 2. Let P be a strict-interval ABP and v any node in P. Then, v is either ascending or descending and not both.

Proof. Let $I = I_{s,v}$ and $J = I_{v,t}$. Since P is a strict-interval ABP, the intervals I and J are disjoint and hence either $I \preceq J$ or $J \preceq I$ as required. $\square$

Consider any s to t path ρ in P. We say that ρ is *ascending* if every node in ρ except s and t is ascending. Similarly, ρ is called descending if every node in ρ except s and t is descending.

Lemma 4. *Let P be a strict interval ABP and let ρ any s to t path in P. Then either ρ is ascending or descending.*

Proof. We prove that no s to t path in P can have both ascending and descending nodes. For the sake of contradiction, suppose that ρ has both ascending and descending nodes. There are two cases. In the first, there is an edge (u, v) in ρ such that u is an ascending node and v is a descending node. Let $I = I_{s,u}$, $J = I_{u,t}$, $I' = I_{s,v}$ and $J' = I_{v,t}$. Since $P_{s,u}$ is a sub-program of $P_{s,v}$, we have $I \subseteq I'$, similarly $J' \subseteq J$. By the assumption, we have $I \preceq J$ and $J' \preceq I'$. By Observation 1, we have $I \preceq J'$ and $J' \preceq I'$. By transitivity, we have $I \preceq I'$. However, by the definition of $\preceq$, I and I' are incomparable, which is a contradiction. The second possibility is u being a descending node and v being an ascending node. In this case, $J \preceq I$ and $I' \preceq J'$. Then, by Observation 1, we have $J' \preceq I$ as $J' \subseteq J$. Therefore, $J \preceq J'$ by the transitivity of $\preceq$, a contradiction. This completes the proof. $\square$

Lemma 4 implies that the set of all non-terminal nodes of P can be partitioned into two sets such that there is no edge from one set to the other. Formally:

Lemma 5. *Let P be an interval ABP. There exist two strict-interval ABPs P_1 and P_2 such that*

1. *All non-terminal nodes of P_1 are ascending nodes and all non-terminal nodes of P_2 are descending nodes; and*
2. *$P = P_1 + P_2$.*

Proof. Let P_1 be the sub-program of P obtained by removing all descending nodes from P and P_2 be the sub-program of P obtained by removing all ascending nodes in P. By Lemma 4, the non-terminal nodes in P_1 and P_2 are disjoint and every s to t path ρ in P is either a s to t path in P_1 or a s to t path in P_2 but not both. Thus $P = P_1 + P_2$. □

Next we show that any strict-interval ABP consisting only of ascending or only of descending nodes can in fact be converted into an ROABP.

Lemma 6. *Let P be a strict-interval ABP consisting only of ascending nodes or only of descending nodes. Then the polynomial computed by P can also be computed by a ROABP P' of size polynomial in $\mathsf{size}(P)$. The order of variables in P' is $x_1, \ldots, x_n$ if P has only ascending nodes and $x_n, \ldots, x_1$ if P has only descending nodes.*

Proof. We consider the case when all non-terminal nodes of P are ascending nodes. Let ρ be any s to t path in P. We claim that the edge labels in ρ are according to the order $x_1, \ldots, x_n$. Suppose that there are edges (u, v) and (u', v') occurring in that order in ρ such that (u, v) is labelled by x_i and (u', v') is labelled by x_j with $j < i$. Let $I' = I_{s,u'}$ and $J' = I_{u',t}$. Since $i \in I'$, $j \in J'$ and $I' \cap J' = \emptyset$, it must be the case that $J' \preceq I'$ and hence u' must be a descending node, a contradiction. This establishes that P is an one ordered ABP. By the equivalence between one ordered ABPs and ROABPs (Jansen (2008), Jansen *et al.* (2010)), we conclude that the polynomial computed by P can also be computed by a ROABP of size polynomial in the size of P.

The argument is similar when all non-terminal nodes of P are descending. In this case, we have $i < j$ in the above argument and hence $I' \preceq J'$, making u' an ascending node leading to a contradiction. This concludes the proof. □

A permutation π of $[1, n]$ naturally induces the order $x_{\pi(1)}, \ldots, x_{\pi(n)}$. The *reverse* of π is the order $x_{\pi(n)}, x_{\pi(n-1)}, \ldots, x_{\pi(1)}$. Since branching programs are layered, any multilinear polynomial computed by a ROABP where variables occur in the order given by π can also be computed by a ROABP where variables occur in the reverse of π.

Observation 3. Let P be a ROABP where variables occur in the order induced by a permutation π. The polynomial computed by P can also be computed by a ROABP of same size as P that reads variables in the reverse order corresponding to π.

Proof. Let P' be the ROABP obtained by reversing the edges of P and swapping the start and terminal nodes. Since P is a layered DAG, there is a bijection between the set of all s to t paths in P and the set of all s to t paths in P', where the order of occurrence of nodes and hence the edge labels are reversed. This completes the proof. $\square$

The above observations immediately establish Theorem 5.

Proof of Theorem 5. Let P be a strict-interval ABP of size S computing a multilinear polynomial f. By Lemma 5 there are strict interval ABPs P_1 and P_2 such that P_1 has only ascending non-terminal nodes and P_2 has only descending non-terminal nodes such that $f = f_1 + f_2$ where f_i is the polynomial computed by P_i, $i \in \{1, 2\}$. By Lemma 6 and Observation 3, f_1 and f_2 can be computed by a ROABPs that read the variables in the order $x_1, \ldots, x_n$. Then $f_1 + f_2$ can also be computed by an ROABP. It remains to bound the size of the resulting ROABP. Note that $\mathsf{size}(P_i) \leq S$. A ROABP for f_i can be obtained by staggering the reads of P_i which blows up the size of the ABP by a factor of n (Jansen (2008), Jansen *et al.* (2010)). Therefore size of the resulting ROABP is at most $2nS \leq O(S^2)$. $\square$

Using Theorem 5, we can design the following white-box PIT for strict-interval ABPs.

Corollary 1. *Given a strict-interval ABP P of size s, we can check whether the polynomial computed by P is identically zero in time $O(poly(S))$.*

Proof. The proof follows from Theorem 5 and the polynomial time white-box PIT algorithm given by Raz and Shpilka (2005) for non-commutative ABPs, since the variables in X are read only once, in a fixed order, in a ROABP. $\square$

Now, since we observed that reading once in a particular order in an ABP is the same as reading intervals of variables along every path and in every sub-program, it is interesting to inspect if we can obtain good lower bounds against ROABPs. It is known that the class of read-once formulas is contained in the class of ROABPs. It is interesting to investigate whether ROABPs have more computational power than a sum of read-once formulas.

3.6 Sum of ROFs

Ramya and Rao (2019*b*) show an exponential lower bound on the sum of read-once formulas computing the Raz-Yehudayoff polynomial, that can be efficiently computed by multilinear circuits (Raz and Yehudayoff (2008)). Kayal *et al.* (2016) show a separation between ROABPs and depth-3 multilinear circuits. In this context, the relation between the classes of ROABPs and the sum of ROFs model is necessary to complete a part of the picture.

In this section, we show an exponential lower bound on the sum of ROFs against a hard polynomial that is efficiently computable by a ROABP. This hard polynomial is defined as the product of Raz-Yehudayoff polynomials on disjoint partitions of the set of input variables, X.

3.6.1 A Hard Polynomial

Let $X = \{x_1, \ldots, x_n\}$ be the set of input variables of the hard polynomial such that $4 \mid n$. The Raz-Yehudayoff polynomial f_{RY} (Raz and Yehudayoff (2008)) is defined on a set of even number of variables where there is a natural ordering on the indices of the variables. We revisit the definition of this polynomial as stated in Chapter 2. Given that an interval $\{a \mid a, i, j \in \mathbb{N}, i \leq a \leq j\}$ is denoted by the notation $[i, j]$, and the sets of variables $X = \{x_1, \ldots, x_{2m}\}$, $W = \{w_{i,\ell,j}\}_{i,\ell,j \in [2m]}$, the Raz-Yehudayoff polynomial is as follows.

Definition 13. (Raz and Yehudayoff (2008)) Let us consider $f_{ij} \in \mathbb{F}[X, W]$ defined over the interval $[i, j]$. If the length of the interval $[i, j]$, $|[i, j]| = 1$, $f_{ij} = 0$. For

$|[i,j]| > 0$,

$$f_{ij} = (1 + x_i x_j) f_{i+1,j-1} + \sum_{\ell \in [i+1,j-2]} w_{i,\ell,j} f_{i,\ell} f_{\ell+1,j},$$

where we assume without loss of generality, lengths of $[i,\ell],\ [\ell+1,j]$ are even and smaller than $[i,j]$. We define $f_{1,2m}$ as the Raz-Yehudayoff polynomial $f_{\mathsf{RY}}(X)$.

We use $f_{\mathsf{RY}}(X')$ to denote the Raz-Yehudayoff polynomial defined on the variable set X' of even size, where X' is an arbitrary subset of X.

Let $r = \Theta(1)$ be an integer factor of n such that r and n/r are both even. For $1 \leq i \leq n/r$, let B_i be the variable set $\{x_{(i-1)r+1}, \ldots, x_{ir}\}$ and $\mathcal{B}$ denote the partition $B_1 \cup B_2 \cup \cdots \cup B_{n/r}$ of X. The hard polynomial f_{PRY} is defined as follows.

$$f_{\mathsf{PRY}} = f_{\mathsf{RY}}(B_1) \cdot f_{\mathsf{RY}}(B_2) \cdots f_{\mathsf{RY}}(B_{n/r}). \tag{3.1}$$

By definition of the polynomial f_{PRY}, it can be computed by a constant-width ROABP of polynomial size as well as by a polynomial size depth three $\Pi\Sigma\Pi$ syntactic multilinear circuit.

In order to prove a lower bound against a class of circuits computing the polynomial f_{PRY}, we consider the complexity measure of the rank of partial derivative matrix. Like in Raz (2006) and many follow-up results, we analyse the rank of the partial derivative matrix of f_{PRY} under a random partition. The reader might have already noticed that there are equi-partitions under which the $\mathsf{rank}_\varphi(f_{\mathsf{PRY}}) = 1$. Thus, we need a different distribution on the equi-partitions under which f_{PRY} has full rank with probability 1. In fact, under any partition φ, which induces an equi-partition on each of the variable blocks B_i, we have $\mathsf{rank}_\varphi(f_{\mathsf{PRY}}) = 2^{n/2}$, i.e., full rank. We define $\mathcal{D}_B$ as the uniform distribution on all such partitions. Formally, we have:

Definition 20. (Distribution $\mathcal{D}_{\mathcal{B}}$) The distribution $\mathcal{D}_{\mathcal{B}}$ is the distribution on the set of all equi-partitions $\hat{\varphi}$ of X obtained by independently sampling an equi-partition φ_i of each variable blocks B_i, for all i such that $1 \leq i \leq n/r$. We express $\hat{\varphi}$ as $\hat{\varphi} = \varphi_1 \circ \ldots \circ \varphi_{n/r}$.

For any partition φ in the support of $\mathcal{D}_{\mathcal{B}}$, we argue that the polynomial f_{PRY} has full rank:

Observation 4. For any $\varphi \sim \mathcal{D}_{\mathcal{B}}$, $\mathsf{rank}_\varphi(f_{\mathsf{PRY}}) = 2^{n/2}$ with probability 1.

Proof. Let us fix an equi-partition function $\hat{\varphi} \sim \mathcal{D}_{\mathcal{B}}$, $\hat{\varphi} : X \to Y \cup Z$. Let $t = r$. Considering $f_{\mathsf{RY}}(X')$ where $|X'| = t$ and t is even, we can prove the partial derivative matrix of $f_{\mathsf{RY}}(X')$ has rank $2^{n/2}$ under $\hat{\varphi}$ by induction on t. By definition of f_{RY}, for $t = 2$ we have $f_{\mathsf{RY}} = 0$.

So, for the higher values of t, we see the term $(1 + x_1x_t)$ and $f_{2,t-1}$ are variable disjoint, where $(1 + x_1x_t)$ has rank ≤ 2, and by the induction hypothesis, $f_{2,t-1}$ has rank $2^{t/2-1}$. Also, by induction hypothesis, for any ℓ, the ranks of partial derivative matrices of $f_{1,\ell}$ and $f_{\ell+1,t}$ are $2^{\ell/2}$ and $2^{(t-\ell)/2}$ respectively.

When $\hat{\varphi}(x_1) \in Y$ and $\hat{\varphi}(x_t) \in Z$, we set $w_{1,\ell,t} = 0$ for all $\ell \in [2, t-1]$ and $\mathsf{rank}_{\hat{\varphi}}(f_{1,t}) = \mathsf{rank}_{\hat{\varphi}}(1 + x_1x_t) \cdot \mathsf{rank}_{\hat{\varphi}} f_{2,t-1} = 2 \cdot 2^{(t/2-1)} = 2^{t/2}$. When $\hat{\varphi}(x_1) \in Y$ and $\hat{\varphi}(x_t) \in Y$, for an arbitrary $\ell \in [t]$ we set $w_{1,\ell,t} = 1$ and we have $\mathsf{rank}_{\hat{\varphi}}(f_{1,t}) = \mathsf{rank}_{\hat{\varphi}}(f_{1,\ell}) \cdot \mathsf{rank}_{\hat{\varphi}}(f_{\ell+1,t}) = 2^{t/2}$, since $\hat{\varphi}$ is an equi-partition.

By sub-additivity of rank, and since $B_i,\ i \in [n/r]$ are disjoint sets of variables, we have $\mathsf{rank}_{\hat{\varphi}}(f_{\mathsf{PRY}}) = \prod_{i\in[n/r]} \mathsf{rank}_{\hat{\varphi}}(f_{\mathsf{RY}}(B_i)) = \prod_{i\in[n/r]} 2^{t/2} = 2^{tn/2r} = 2^{n/2}$. $\square$

3.6.2 Rank Upper Bound on ROFs

In the following, we argue that the polynomial h cannot be computed by sum of ROFs of sub-exponential size. More formally,

Theorem 6. *Let $f_1, \ldots, f_s$ be read-once polynomials such that $f_{\mathsf{PRY}} = f_1 + f_2 + \cdots + f_s$, then $s = 2^{\Omega(n)}$.*

We use the method of obtaining an upper bound on the rank of partial derivative matrix for ROFs with respect to a random partition developed by Ramya and Rao (2019*b*). Though the argument in Ramya and Rao (2019*b*) works for an equi-partition sampled uniformly at random, we show their structural analysis of ROFs can be extended to the case of our distribution $\mathcal{D}_{\mathcal{B}}$. We begin with the notations used in Ramya and Rao (2019*b*) for the categorisation of the gates in a read-once formula F. (In this categorisation, the authors have only considered gates with at least one input being a variable.)

- Type- A: These are sum gates in F with both inputs variables in X.
- Type- B: Product gates in F with both inputs variables in X.

- Type- C: Sum gates in F where only one input is a variable in X.
- Type- D: Product gates in F where only one input is a variable in X.

Thus, type-D gates compute polynomials of the form $h \cdot x_i$ where $x_i \in X, h \in \mathbb{F}[X \setminus \{x_i\}]$ are the inputs to the type-D gate. Let a, b, c, d be the number of gates of type-A, B, C and D respectively. Let a'' be the number of Type A gates that compute a polynomial of rank 2 under an equi-partition φ, and a' be the number of Type-A gates that compute a polynomial of rank 1 under φ such that $a = a' + a''$.

The following lemma is an adaptation, for our distribution $\mathcal{D}_{\mathcal{B}}$, of the same lemma for the distribution of all equi-partitions on n variables from Ramya and Rao (2019*b*).

Lemma 7. *Let $f \in \mathbb{F}[X]$ be an ROP, and φ be an equi-partition function sampled uniformly at random from the distribution $\mathcal{D}_{\mathcal{B}}$. Then with probability at least $1-2^{-\Omega(n)}$,* $\mathsf{rank}_{\varphi}(M_f) \leq 2^{n/2-\Omega(n)}$.

Proof. We first argue a rank upper bound for an arbitrary f_i. Let Φ_i be the formula computing f_i with gates of the types described as above. Let $\hat{\varphi} = \varphi_1 \circ \ldots \circ \varphi_{n/r}$ sampled from the distribution $\mathcal{D}_B$ uniformly at random.

We use the Lemma 3.1 from Ramya and Rao (2019*b*) which concludes that type-D gates do not contribute to the rank of a ROF.

Lemma 8. *(Ramya and Rao, 2019*b*, Lemma 3.1) Let F be a ROF computing a read-once polynomial f and $\varphi : X \to Y \cup Z$ be an partition function on n variables. Then,* $\mathsf{rank}_{\varphi}(f) \leq 2^{a''+\frac{2a'}{3}+\frac{2b}{3}+\frac{9c}{20}}$.

Intuitively, Lemma 8 can be applied to a ROF F under a distribution $\hat{\varphi} \sim \mathcal{D}_B$ as follows. If there are a large number of type D gates (say αn, for some $0 \leq \alpha < 1$), then for any such equi-partition $\hat{\varphi}$, $\mathsf{rank}_{\hat{\varphi}}(f) \leq 2^{(1-\alpha)n/2}$. A type C gate, too, contributes a small value (at most 2) to the rank compared to gates of types A and B. Thus, without loss of generality, we assume that the number of type C and D gates is at most αn. Now our analysis proceeds as in Ramya and Rao (2019*b*), only differing in the estimation of a'', a' under an equi-partition $\hat{\varphi} \sim \mathcal{D}_B$.

Let $(P_1, \ldots, P_t)$ be a pairing induced by the gates of types A and B (i.e., the two inputs to a gate of type A or B form a pair). There can be at most $n/2$ pairs, but since we have αn gates of type C and D for some $0 \leq \alpha < 1$, we assume $(1-\alpha)n$ remaining type A and B gates. Therefore, for $t = (n - \alpha n)/2,\ t \leq n/2$, we have the pairs $P_1, \ldots, P_t$ induced by the type A and B gates in Φ_i.

Now, considering the division of X into $B_1, \ldots, B_{n/r}$, we can divide the pairs into two sets depending on whether a pair lies entirely within a block $B_i,\ i \in [n/r]$ or the pair has its members in two different blocks B_i and B_j for $i, j \in [n/r],\ i \neq j$. We define these two sets as $W = \{P_i \mid P_i = (x, y), \exists \ell,\ x, y \in B_\ell\}$ for pairs lying within blocks and $A = \{P_i \mid P_i = (x, y), \exists j, k,\ j \neq k,\ x \in B_j, y \in B_k\}$ for pairs lying across blocks, where x, y are two arbitrary variables in X.

Each pair P_i can be monochromatic or bichromatic under the randomly sampled equi-partition $\hat{\varphi}$ with the probability $\frac{1}{2}$. Presence of monochromatic edges will give us a reduction in the rank of f_i under $\hat{\varphi}$. The analysis on W and A is done separately as follows.

Analysing W, $|W| > t/2$: Let $B_{i_1}, \ldots, B_{i_\ell}$ be the blocks containing at least one pair from W, $\ell \leq n/r$. We want to estimate ℓ and count how many of these ℓ blocks have at least one monochromatic pair under $\hat{\varphi}$ from W.

For each $B_i,\ i \in [t]$, we define the Bernoulli random variable X_i such that,

$$X_i = \begin{cases} 1, \text{ if } \exists P \in W,\ P = (x, y),\ x, y \in B_i, \\ 0, \text{ otherwise.} \end{cases}$$

Let $\mathsf{Pr}[X_i = 1] = \mathsf{Pr}[\exists P \in W,\ P = (x, y),\ x, y \in B_i] = \epsilon$, for some $\epsilon > 0$.

Then we have $\mathsf{E}[X_i] = \epsilon$, and for $\mathcal{X} = X_1 + \ldots + X_{n/r}$, $\mathsf{E}[\mathcal{X}] = \epsilon \cdot n/r$. By the Chernoff's bound defined in Mitzenmacher and Upfal (2005), we have,

$$\mathsf{Pr}[\mathcal{X} > 2\epsilon n/r] < \exp(\frac{-\epsilon n}{3r}).$$

Now we estimate ϵ as follows:

$$\begin{aligned}\epsilon &= \mathsf{Pr}[X_i = 1] = \mathsf{Pr}[\exists P \in W,\ P = (x,y),\ x,y \in B_i] \\ &= \mathsf{Pr}[x,y \in B_i | \exists P \in W,\ P = (x,y)] \\ &= \frac{\mathsf{Pr}[x,y \in B_i]}{\mathsf{Pr}[\exists P \in W,\ P = (x,y)]} \\ &\geq \mathsf{Pr}[x,y \in B_i] \ \text{ since } \mathsf{Pr}[\exists P \in W,\ P = (x,y)] \leq 1 \\ &= \frac{1}{r^2}.\end{aligned}$$

Therefore, $\mathsf{Pr}[\mathcal{X} > 2\epsilon n/r] < \exp(\frac{-\epsilon n}{3r}) \leq \exp(-\Omega(n))$, when r is a constant. This implies that at least $2/r^2$ fraction of the blocks have a pair entirely within them with probability $1 - \exp(-\Omega(n))$ and each of these pairs is monochromatic under $\hat{\varphi}$ with the constant probability $1/2$. This gives an upper bound on the rank of f_i,

$$\text{rank}_{\hat{\varphi}}(f_i) \leq 2^{n/2 - n/r^3} = 2^{n/2 - \Omega(n)}.$$

Analysing A**,** $|A| > t/2$**:** Since each pair of variables in A lies across two blocks, we create a graph $G = (V, E)$ where each $v_i \in V$ represents the block B_i and $E = \{(v_i, v_j) \mid (x,\ y) \in A,\ x \in B_i,\ y \in B_j, i \neq j\}$.

The graph G has maximum degree r since there can be at most r pairs with one member in a fixed block B_i. If the edges in E form a perfect matching M' in G, then under $\hat{\varphi}$, the edges in E can be either bichromatic or monochromatic. We need to show there will be sufficient number of monochromatic edges to give a tight upper bound for $\text{rank}_{\hat{\varphi}}(f_i)$.

By a result in Biedl *et al.* (2004), any graph with maximum degree r has a maximal matching of size $m/(2r-1)$, where $|E| = m$. Since $|A| \geq t/2$, $m \geq t/2$ and hence the maximal matching is of size $t/2(2r-1) = \Omega(n)$ when r is a suitable constant. With probability $1/2$, an edge in the maximal matching is bichromatic. Hence, $\leq t/2$ number of the edges in the maximal matching are bichromatic with probability $1/2^{t/2} = O(\exp(n^{-1}))$. So, with the high probability of $1 - O(\exp(n^{-1}))$, more than half of the edges in the maximal matching are monochromatic, thus giving us the rank bound,

$$\text{rank}_{\hat{\varphi}}(f_i) \geq 2^{n/2 - t/2} = 2^{n/2 - \Omega(n)}.$$

□

Given an upper bound on the rank of ROFs under a random partition from $\mathcal{D}_{\mathcal{B}}$, we now proceed to prove the Theorem 6 by showing a lower bound on the size of ROFs computing our hard polynomial h.

Proof. (Proof of Theorem 6) By Observation 4, the upper bound on the rank of ROFs given by Lemma 7 and the sub-additivity of rank, we have:

$$s \cdot 2^{n/2-\Omega(n)} \leq 2^{n/2} \implies s = 2^{\Omega(n)}.$$

□

3.7 Conclusion

In this chapter, we study the models of sum of ROABPs and sum of ROFs and obtain lower bounds against them. From the lower bound against the sum of ROABPs model, it is clear that the sum of ROABPs model is weaker than the syntactically multilinear formulas, since Dvir *et al.* (2012) obtain a super-polynomial ($n^{\Omega(\log n)}$) lower bound on the size of syntactically multilinear formulas computing the same polynomial, against which we show a lower bound of $2^{\omega(n)}$ on the size of the sum of ROABPs computing it. It would be interesting to see if a similar or larger lower bound can be obtained against read-k formulas, for some $k > 1$, computing the hard polynomial defined by Dvir *et al.* (2012).

The characterisation of strict-interval ABPs by a ROABP raises the question if a similar characterisation of interval ABPs can be obtained in the form of a sum of ROABPs. Similarly, it is not known if an interval formula can be expressed as a sum of ROFs.

The lower bound on the sum of ROFs against ROABPs motivates the question of a lower bound on sum of ROFs computing a polynomial efficiently computable by read-k formulas for some $k > 1$.

Though these questions, if answered, will give us a clearer view of the landscape of sub-classes of multilinear circuits, we are interested in generalizing the lower bound problem by considering it from a parameterized point of view. For this purpose, we first consider the possibility of efficient depth reduction results for parameterized arithmetic circuits, which we consider the following chapter.

CHAPTER 4

PARAMETERIZED DEPTH REDUCTION AND LOWER BOUNDS

4.1 Introduction

The problem of depth reduction is to find the minimum depth necessary in an arithmetic circuit in order to compute the polynomials in the class VP i.e., polynomials that have polynomial size circuits, by incurring a small blow-up in the size of the circuit. Such a result is important since the depth of an arithmetic circuit C computing a polynomial f represents the maximum amount of sequential computation necessary to compute f. The first depth reduction result was by Valiant *et al.* (1983), who showed that only a depth of $O(\log n)$ is necessary to compute all polynomials in VP. More formally, we have:

Proposition 3. (Valiant *et al.* (1983)) If a homogeneous polynomial f of degree d is computed by a circuit of size s, then there exists a homogeneous circuit computing f, of size poly(s, d) and depth $O(\log d)$, such that add gates in the circuit are allowed unbounded fan-in, and product gates have fan-in 5.

This implies that for every arithmetic circuit C of arbitrary depth and polynomial size there is a circuit C' of depth bounded by $O(\log n)$ (assuming degree $d = n^{O(1)}$) having the same computational power, such that C' is not much larger in size. The method of obtaining this seminal result is simple and elegant. The circuit C is viewed in two parts i.e., C is cut at degree $d/2$ into a top half C_t having gates computing polynomials of degree $\geq d/2$ as leaves, and a bottom half, C_b. The polynomial computed by the circuit C, say f, is expressed as a small sum of products of polynomials computed by the leaves of C_t by bounding the fan-in of product gates.

As the polynomial f of degree d is now represented by a sum of product of degree $d/2$ polynomials, the authors repeat the same process with the degree $d/2$ polynomials and so on. Thus, they are able to obtain a recurrence relation that expresses depth of a circuit with parameter d as a linear function of the depth of a circuit with parameter $d/2$. Therefore, the circuit C can be parallelized to a depth logarithmic in d. The construction of the depth-reduced circuit C' also fixes the fan-in of each product gate to the constant 5 and preserves the homogeneity of the original circuit C. This result implies that the task of proving arithmetic circuit lower bounds against arithmetic circuits is reduced to proving lower bounds against the class of $O(\log d)$ depth circuits.

The result by Valiant *et al.* (1983) was followed by a number of lower bound results against constant-depth circuits. The earliest such result was the super-polynomial lower bound shown by Nisan and Wigderson against depth-3 circuits, using the measure of dimension of the space spanned by partial derivatives of the given class of polynomials. This was followed by the lower bounds against depth-3 circuits shown by Grigoriev and Karpinski (1998) and later on, Grigoriev and Razborov (2000), against computing the permanent and determinant polynomials. Both of these works used the dimension of partial derivatives as the measure.

Following the lower bound results, there were multiple results in construction of efficient deterministic PIT algorithms. The first such work was the black-boxPIT algorithm given by Klivans and Spielman (2001) for sparse polynomials, i.e., polynomials computed by depth-2 circuits of polynomial size. Dvir and Shpilka (2007) gave a deterministic quasi-polynomial time PIT for restricted depth-3 circuits of bounded top fan-in. In Dvir and Shpilka (2007), the restriction on the depth-3 circuit was a bound on the rank of the linear forms computed by the depth-2 gates. Their result was subsequently improved by Kayal and Saxena (2007), Saxena and Seshadhri (2010).

The literature on constant-depth circuits implied that obtaining efficient PIT algorithms and strong lower bounds against circuits of bounded depth might be difficult. Hence, it could be safely assumed that these classes of constant-depth circuits held considerable computational power. This insight motivated Agrawal and Vinay (2008) to study if depth of a given unrestricted arithmetic circuit could be reduced to a constant, and they discovered that circuits of depth-4 and sub-exponential size indeed had the same computational power as circuits of unbounded depth and polynomial size. Their result is stated formally as follows.

Proposition 4 (Agrawal and Vinay (2008))**.** If a n-variate polynomial p of degree d

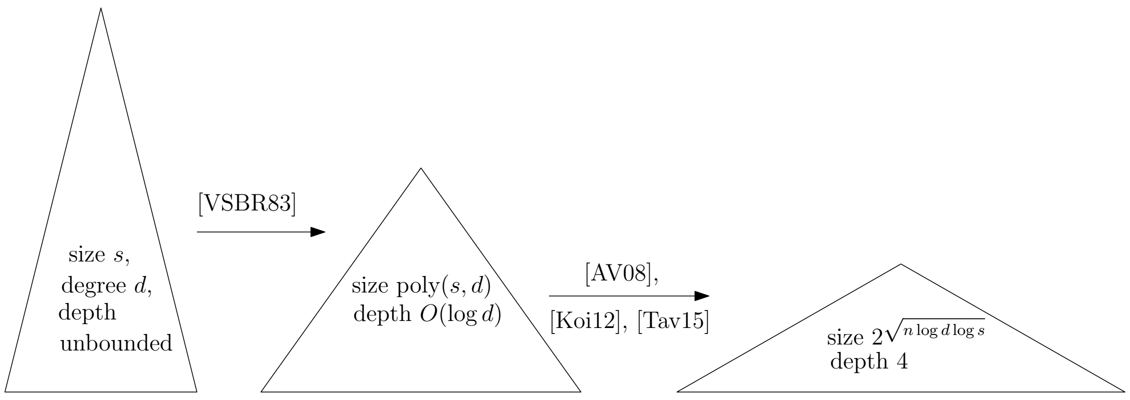


Figure 4.1: Visualisation of depth reduction results.

is computed by an arithmetic circuit C of size $2^{o(n)}$ and unbounded depth, then there exists a circuit C' of size $2^{o(n)}$ and depth-4 computing it.

The result by Agrawal and Vinay (2008) implies that for any arithmetic circuit C of polynomial size, the equivalent depth-4 circuit C' is of sub-exponential size. Hence, with a larger blow-up in size than the construction given by Valiant *et al.* (1983), but a constant depth of 4, the depth-reduced circuit C' has the same computational power as an unbounded depth circuit. This blow-up in size was subsequently reduced by Koiran (2012); followed by Tavenas (2015) who reduced the size of C' to $2^{O(\sqrt{n}\log n)}$. These depth-reduction results motivated further research in proving lower bounds against depth-4 circuits, since proving the Valiant's hypothesis would require proving exponential lower bounds against circuits of depth-4 computing the permanent polynomial. Hence, the study of lower bounds against constant-depth circuits computing hard polynomials follow from the study of depth reduction.

In this chapter, our aim is to study depth reduction in the parameterized setting. Depth reduction involves obtaining a small upper bound on the size of the circuit of reduced depth constructed from the unbounded depth circuit. Since the parameterized notion of efficiency in size is relaxed by a factor of a function on the parameter compared to the classical setting, it seems interesting to investigate if the depth reduction results in the classical setting translate to circuits where the size is expressed in terms of both the size and degree of the polynomial computed by it.

As the question of depth reduction is closely related to the arithmetic circuit lower bounds problem, we also show lower bounds against circuits of depth-4. For this purpose, we use the complexity measure of dimension of partial derivatives defined by

Nisan (1991).

We observe that the depth reduction result by Valiant *et al.* (1983) translates to arithmetic circuits parameterized be degree, but the result by Agrawal and Vinay (2008) does not. This is because a lower bound of $n^{O(k)}$ can be obtained on the size of depth-4 circuits with restricted top product gate fan-in computing an explicit polynomial that can be efficiently computed by a depth-4 circuit of top product gate fan-in bounded only by k, the degree.

4.2 Chapter Outline

In Section 4.3, the possibility of parameterized depth reduction is discussed. A parameterized version of the depth reduction by Valiant *et al.* (1983) is stated as follows.

Proposition 5. Valiant *et al.* (1983) Any degree-parameterized polynomial family (p, k) computable by a circuit of FPT size can be computed by a circuit of depth $g(k)\log n$ and size $g'(k)n^{O(1)}$, for some functions g and g' that depend only on the parameter.

In Section 4.4, we obtain a $n^{\Omega(k)}$ lower bound against depth-5 powering circuits of bounded top product gate fan-in.

Theorem 7. *There is a polynomial* p *computed by a* $\Pi^{k/2}\Sigma\wedge^2$ *circuit of polynomial size such that any* $\Sigma \wedge^\alpha \Sigma \wedge^d \Sigma$ *circuit computing* p *has size* $n^{\Omega(k)}$*, for any* $\alpha = o(k)$ *and* $d = k/\alpha$*.*

This rules out a parameterized version of the depth reduction by Agrawal and Vinay (2008) as a consequence.

Corollary 2. *There is a parameterized family of polynomials that can be computed by depth four circuits of polynomial size, but any depth four* $\Sigma\Pi^{o(k)}\Sigma\Pi^k$ *circuit computing it requires size* $n^{\Omega(k)}$*.*

4.3 Parameterized Depth Reduction

The question of depth reduction is one of the most fundamental structural aspects in algebraic complexity theory, and can be formally expressed as follows.

Given a polynomial family $p = (p_n)_{n\geq 0}$ and a size bound $s = s(n)$, what is the minimum depth of an arithmetic circuit of size s computing p?

This question can be re-stated in the parameterized context as the *parameterized depth reduction problem*, which is formally defined as follows.

Given a parameterized polynomial family (p, k) in FPT what is the minimum depth of a size $g(k)n^c$ circuit computing p where $g(k)$ is an arbitrary function of k and c is some constant?

By applying the depth reduction result by Valiant *et al.* (1983) (Proposition 3) on circuits parameterized by degree having FPT size, we state a parameterized version of their depth reduction result as follows.

Proposition 5. Valiant *et al.* (1983) Any degree-parameterized polynomial family (p, k) computable by a circuit of FPT size can be computed by a circuit of depth $g(k)\log n$ and size $g'(k)n^{O(1)}$, for some functions g and g' that depend only on the parameter.

In the surprising result by Agrawal and Vinay (2008) (Proposition 4), the authors showed that any homogeneous polynomial f computed by polynomial size arithmetic circuits can be computed by depth four $\Sigma\Pi^{\sqrt{n}}\Sigma\Pi^{\sqrt{n}}$ homogeneous circuits of size $2^{o(n)}$ (subsequently improved by Tavenas (2015) to $2^{\sqrt{n}\log n}$). Later, Gupta *et al.* (2014) proved that over infinite fields, there is a depth three $\Sigma\Pi\Sigma$ circuit of size $2^{\sqrt{n}\log n}$ for f.

A parameterized counterpart of the depth reduction in Agrawal and Vinay (2008) would be to transform a circuit C of size $g(k)n^{O(1)}$ and syntactic degree k to a depth four $\Sigma\Pi\Sigma\Pi$ circuit of syntactic degree k and size $g'(k)n^{O(1)}$ where g and g' are functions of k alone. Note that a $\Sigma\Pi\Sigma\Pi$ circuit C of syntactic degree k will have Π fan-in bounded by k at both of the Π layers. So we can assume it to be of the form $\Sigma\Pi^k\Sigma\Pi^k$. Further, if C is homogeneous with the bottom Π layer having syntactic degree t then C can be assumed to be a homogeneous $\Sigma\Pi^{k/t}\Sigma\Pi^t$ circuit.

We first observe that we can replace Π gates with $\wedge$ (powering) gates in any depth four circuit with syntactic degree bounded by the parameter k. This is because on applying Fischer's identity, for any polynomial computed by a $\Sigma\Pi^a\Sigma\Pi^b$ circuit of size s, an equivalent $\Sigma\wedge^a\Sigma\wedge^b\Sigma$ powering circuit can be obtained such that the depth-5 powering circuit has size $\mathsf{poly}(s)\cdot 2^{\max\{a,b\}}$. The proof of this statement can be seen in Gupta *et al.* (2014). The following is a re-statement of the same, in the parameterized context where $a = k/t$, $b = t$.

Lemma 9. *Let C be a $\Sigma\Pi^{k/t}\Sigma\Pi^t$ circuit of size s computing a polynomial p over $\mathbb{Z}$. Then there is a $\Sigma\wedge^{k/t}\Sigma\wedge^t\Sigma$ circuit C' of size $\max\{2^{k/t}, 2^t\}\cdot s$ computing p. Moreover, if C is homogeneous, so is C'.*

Thus a parameterized version of depth reduction in Agrawal and Vinay (2008) would imply that every parameterized polynomial family (p, k) in FPT can be computed by a homogeneous $\Sigma\wedge^{O(\sqrt{k})}\Sigma\wedge^{O(\sqrt{k})}\Sigma$ circuit of size $g(k)n^{O(1)}$ for some function g of k. However, in the next section, we show that if the degree of top layer of powering gates is bounded by $o(k)$, then this is not possible.

4.4 Parameterized Lower Bound on Depth-5 Powering Circuits

In this section we show that among parameterized depth-5 powering circuits, that are equivalent to depth-4 circuits by a simple application of Fischer's identity Fischer (1994), a top fan-in of $O(k)$ gives considerably more computational power than a top fan-in of $o(k)$.

We consider the depth-5 powering circuit to be of the form $\Sigma\wedge^{O(\frac{k}{t})}\Sigma\wedge^t\Sigma$. By Lemma 9, we know that $\Sigma\Pi^{O(\frac{k}{t})}\Sigma\Pi^t$ circuit of size s can be transformed into a $\Sigma\wedge^{O(\frac{k}{t})}$ $\Sigma\wedge^t\Sigma$ circuit of size $s2^{\max\{k/t,t\}}$. In fact Lemma 9 holds for any chosen $t \leq k$. We show that there is a polynomial computable by $\Pi^k\Sigma\wedge^2$ circuits of polynomial size that cannot be computed by a $\Sigma\wedge^{o(k)}\Sigma\wedge^k\Sigma$ circuit of size $g(k)n^{O(1)}$. Since $\Pi^k\Sigma\wedge^2$ circuits are a sub-class of $\Sigma\Pi^k\Sigma\Pi^2$ circuits by Fischer (1994), our inference regarding the depth-4 circuits of large top product gate fan-in having greater computational power holds.

Theorem 7. *There is a polynomial p computed by a $\Pi^{k/2}\Sigma\wedge^2$ circuit of polynomial size such that any $\Sigma\wedge^\alpha\Sigma\wedge^d\Sigma$ circuit computing p has size $n^{\Omega(k)}$, for any $\alpha = o(k)$ and $d = k/\alpha$.*

A polynomial $f \in \mathbb{F}[X]$ computed by a depth-3 powering circuit (also denoted by depth-3 diagonal circuit in Saxena (2008)) is of the form $f = \ell_1^d + \ell_2^d + \cdots + \ell_t^d$ where each ℓ_i is a linear form in n variables, $t = g(k)\mathsf{poly}(n)$ for some computable $g : \mathbb{N} \to \mathbb{N}$ and d is the powering gate fan-in.

We consider the space spanned by k^{th} order partial derivatives of the polynomial f, $\partial^{\leq k}(f)$. We may observe that for $r < k$, the r^{th} order partial derivatives of ℓ^k are all multiples of ℓ^{k-r}, where ℓ is a linear form. We use $\dim(S)$ to denote the dimension of the space spanned by polynomials in a set S of polynomials in $\mathbb{F}[X]$. We show that Theorem 7 follows immediately from Lemma 10 and 11.

Lemma 10. *Let $\alpha = o(k)$ and $f = \ell_1^d + \ell_2^d + \cdots + \ell_t^d$ where $\ell_i,\ i \in [t]$ are linear forms in the variables in X, $d \leq k$ and $t = g(k)n^{O(1)}$. Then,*

$$dim(\langle\partial^{\leq r} f^\alpha\rangle) \leq g'(k)n^{o(k)},$$

for some computable function g', $\alpha < r$ and $r = o(k)$.

Proof. By definition of partial derivatives of order $i,\ i \in [r]$ of a polynomial f^α, where $\alpha < r$ and $r < k$, we know each polynomial in this set is $f^{\alpha-i}$ times the order i partial derivative of f. Applying the standard rules for computing partial derivatives, we conclude that the space spanned by partial derivatives of order $\leq r$ of the polynomial take the following form.

$$\langle\partial^{\leq r} f^\alpha\rangle \subseteq \mathbb{F} - \text{span}\left\{f^{\alpha-i} \odot \left(\ell_{j_1}^{d-r_1} \cdot \ldots \cdot \ell_{j_i}^{d-r_i}\right) \;\middle|\; i \in [r], {}^{r_1+\ldots+r_i=r}_{j_1,\ldots,j_i\in[t]}\right\}, \tag{4.1}$$

where the operation $\odot$ denotes the product between the polynomials in the set $\{f^{\alpha-i} \mid i \in [r]\}$ and the set of polynomials $\{\ell_{j_1}^{d-r_1} \cdot \ldots \cdot \ell_{j_i}^{d-r_i} \mid r_1 + \ldots + r_i = i, j_1, \ldots, j_i \in [t]\}$.

Analysing the equation 4.1, we conclude the following. There can be at most i^i partitions of i into $r_1, \ldots, r_i$, and since $i \leq r$, this quantity can be upper bounded by r^i.

There are $\binom{t}{i}$ linear terms whose powers add up to $r(d-1)$.

Hence the dimension of $\mathbb{F}-\text{span}(\langle \partial^{\leq r} f^{\alpha}\rangle)$ is bounded by $\sum_{i=1}^{\alpha}\binom{t}{i}r^i \leq k\binom{t}{\alpha}k^{\alpha}$. Given that $t = g(k)n^c$ for some $c > 0$ and g a function of k, we get $\dim(\mathbb{F}-\text{span}(\langle \partial^{\leq r} f^{\alpha}\rangle)) \leq g'(k)n^{o(k)}$, where $g'(k) = kg(k)^k$. □

The above result gives an upper bound on the dimension of partial derivatives measure for a $\Sigma^s \bigwedge^{\alpha} \Sigma^t \bigwedge^{d} \Sigma$ circuit as $s \cdot g'(k)n^{o(k)}$. Now, we define an explicit parameterized polynomial of degree k computable by a $\Pi^{O(k)}\Sigma\Pi^2$ circuit of FPT size such that it exhibits a large lower bound on the dimension of partial derivatives measure.

Lemma 11. *There is a polynomial $p \in \mathbb{F}[x_1, \ldots, x_n]$ of degree k that con be computed by polynomial size $\Pi^{O(k)}\Sigma\wedge^2$ circuits with $\dim(\partial^{\leq k/2}p) = n^{\Omega(k)}$.*

Proof. The polynomial p is defined as follows:

$$p = (x_1^2 + \cdots + x_{\frac{2n}{k}}^2) \cdot \cdots \cdot (x_{\frac{2(k-1)n}{k}+1}^2 + \cdots + x_n^2).$$

Let $B_i \subseteq \{x_1, \ldots, x_n\}$ such that $B_i = \{x_{\frac{2n(i-1)}{k}+1}, \ldots, x_{\frac{2ni}{k}}\}, \forall i \in \{1, \ldots, \frac{k}{2}\}$.

Let $T \subset \{x_1, \ldots, x_n\}$, with $|T| = r$. If $\exists x_p, x_q \in T$ such that $\exists i, x_p, x_q \in B_i$, then $\frac{\partial^r p}{\partial T} = 0$. Otherwise, T contains exactly one variable from r choices of B_is, $r < \frac{k}{2}$, hence:

$$\frac{\partial^r p}{\partial T} = c_T \prod_{x_t \in T} x_t \prod_{j=1}^{\frac{k}{2}-r} p_{i_j},$$

where c_T is a constant, i_js are indices of blocks B_{i_j} that do not contain any variables in T and $p_\ell = x_{\frac{2n}{k}(\ell-1)+1}^2 + \ldots + x_{\frac{2n\ell}{k}}^2$. So for all ℓ such that $T \cap B_\ell = \phi$, p_ℓ divides $\frac{\partial^r p}{\partial T}$.

Therefore, $\dim(\langle \partial^{=r} p\rangle) = \binom{\frac{k}{2}}{r}\left(\frac{2n}{k}\right)^r$. For all $r < \frac{k}{2}$, we can say $\dim(\langle \partial^{\leq r} p\rangle) = \frac{n^{\Omega(k)}}{g(k)}$ for some function g. □

Using Lemma 11 and Lemma 10, we obtain a lower bound on s, the top gate fan-in of a depth-5 powering circuit of low top power gate fan-in $o(k)$, thus proving Theorem 7 as follows.

Proof of Theorem 7. By the construction of the polynomial p in Lemma 11, it is clear that it can be computed by a $\Pi^{k/2}\Sigma\bigwedge^2$ circuit of size $(\frac{2n}{k}+1)\frac{k}{2}+1 = O(n^2)$. Since in the study of arithmetic circuits, depth-3 circuits are considered to be of the form $\Sigma\Pi\Sigma$, we consider p to be efficiently computable by a polynomial size $\Sigma^1\Pi^{k/2}\Sigma\Pi^2$ circuit (a power gate $\wedge$ being a special kind of product gate, we can replace power gates by product gates of the same fan-in).

From Lemma 10, we have the dimension of partial derivatives of $\bigwedge^{o(k)}\Sigma\bigwedge^d\Sigma$ circuits, for any $d \leq k$, to be at most $g'(k)n^{o(k)}$, g' being an arbitrary computable function in k. Then, a $\Sigma^s\bigwedge^{o(k)}\Sigma\bigwedge^d\Sigma$ circuit computing p will have $\dim(\langle\partial^{\leq r}p\rangle) \leq s\cdot g'(k)n^{o(k)}$ by sub-additivity property of the measure. Thus, using Lemma 11, we have a lower bound on s as follows:

$$\frac{n^{\Omega(k)}}{g(k)} \leq \dim(\langle\partial^{\leq r}p\rangle) \leq s\cdot g'(k)n^{o(k)} \implies s = n^{\Omega(k)}.$$

$\square$

Theorem 7 implies that a parameterized version of depth reduction in Agrawal and Vinay (2008) is not possible when the top layer of product gates have fan-in bounded by $o(k)$:

Corollary 2. *There is a parameterized family of polynomials that can be computed by depth four circuits of polynomial size, but any depth four $\Sigma\Pi^{o(k)}\Sigma\Pi^{O(k)}$ circuit computing it requires size $n^{\Omega(k)}$. This implies that a parameterized version of the depth-reduction result by Agrawal and Vinay (2008) is not possible for circuits parameterized by degree.*

Proof. By Fischer's Lemma (Fischer (1994)), any $\Sigma\Pi^{o(k)}\Sigma\Pi^{O(k)}$ circuit can be converted to a $\Sigma\bigwedge^{o(k)}\Sigma\bigwedge^{O(k)}\Sigma$ with a $2^{O(k)}$ blow up in size. Thus the family of polynomials $(p_{n,k})_{n,k\geq 0}$ which can be computed by a $\Pi^{O(k)}\Sigma\Pi$ circuit of polynomial size requires $n^{\Omega(k)}$ size for any $\Sigma\Pi^{o(k)}\Sigma\Pi$ computing it.

As we are unable to obtain a depth-4 degree-k circuit computing p in FPT-size when the fan-in of the top product gate is restricted to $o(k)$, it is clear that depth-4 degree-k circuits of top product gate fan-in $O(k)$ have strictly more computational power than when the top product gate fan-in is bounded by $o(k)$. However, a parameterized version of the statement of the depth-reduction result by Agrawal and Vinay (2008) would entail that depth-4 degree-k circuits, even of top product gate fan-in $o(k)$, be able to compute p in FPT-size, since there is a circuit C_1 computing p in FPT-size. Hence, we conclude that depth-reduction to depth-4 is not possible for circuits parameterized by degree where the product gates have fan-in $o(k)$. □

Remark 1. Note that, in the above, it is necessary that $\alpha = o(k)$. Also, the value of d does not affect the upper bound, in fact, the proof holds even when $d = \Omega(k)$. However, for $\alpha = \Omega(k)$, the above proof is not effective in proving the separation between depth-4 circuits and depth-5 powering circuits.

4.5 Conclusion

Though we are able to show that a depth reduction to depth-4 is not possible for parameterized arithmetic circuits, we cannot rule out the possibility of depth reduction to a constant-depth circuit for a larger value of the constant. The possibility of such a result will be even more evident if we can show a separation between the classes of parameterized arithmetic circuits of depth-4 and depth-5. Since, in Ghosal *et al.* (2017), the authors show a separation between the classes of parameterized depth-3 and depth-4 circuits, such a separation between circuits of higher depths can also be explored.

The lower bound shown against depth-5 powering circuits also motivates further exploration of parameterized lower bounds. Such a direction is interesting because it will give us a view of the parameterized arithmetic circuit landscape and may enable us to develop further techniques for obtaining lower bounds in the non-parameterized setting.

CHAPTER 5

PARAMETERIZED LOWER BOUNDS AGAINST MULTILINEAR ALGEBRAIC CIRCUITS

5.1 Introduction

The lack of progress in proving lower bounds against general arithmetic circuits leads us to consider other perspectives on the problem. The motivation for proving lower bounds against parameterized arithmetic circuits stems from our interest in examining a more generalized view of the lower bound problem, by expressing lower bounds in terms of both the input size and a function of the degree of the polynomial. Such a view might lead to a greater understanding of the difficulties in solving the arithmetic circuits lower bound problem.

We consider multilinear models in our study of parameterized lower bounds not only because they are well-studied but also because multilinear models like read-once formulas and oblivious ABPs exhibit structural properties that can be exploited in order to obtain a lower bound against the size of these models for computing the explicit hard multilinear polynomial.

However, explicit multilinear polynomials defined in the classical setting, like the Raz-Yehudayoff polynomial (Raz and Yehudayoff (2008)) and the DMPY polynomial (Dvir *et al.* (2012)), have degrees of $O(n)$. Thus, these polynomials cannot be parameterized by their degree, since parameters k are always chosen such that $k \ll n$. Therefore, the biggest challenge here is to construct explicit polynomials of degree k such that they exhibit a high value of the complexity measure, and can thus be used to prove lower bounds.

The behaviour of the complexity measure of rank of the partial derivatives matrix, used in Raz and Yehudayoff (2008), is not known under parameterization by the degree of the polynomial. Hence, the notion of *full rank* needs to be defined in the parameterized context so that the hard multilinear polynomial can be designed with the aim of obtaining full rank under any partition, similar to the hard polynomials in the classical setting.

In this chapter, we first define the notion of full rank in the degree-parameterized setting. We, then, show two different constructions of explicit multilinear polynomials which attain close to full rank. While the first polynomial has been obtained using the notion of perfect matchings on graphs, the construction of the second hard polynomial is inspired by the construction of a multilinear hard polynomial in Kayal *et al.* (2016).

We proceed to obtain lower bounds against the multilinear models of read-once oblivious ABPs and strict-interval ABPs. Since our second hard polynomial can be expressed as a sum of three read-once formulas, we obtain a lower bound against sum of read-once formulas by restricting the variables to be read in a fixed order.

5.2 Chapter Outline

In Section 5.3 we construct two degree-k multilinear polynomials. The first polynomial (defined in Section 5.3.1) attains full rank.

Theorem 8. *For the parameterized multilinear polynomial family $f = (f_{n,2k})_{n,k\geq 0}$ such that $f(X) = \sum_{M\in\mathcal{M}} \zeta_M \prod_{(i,j)\sim M}(1+p_{ij})$, we have,*

$$\mathsf{rank}_{\varphi}(f_{n,2k}) = \Omega\left(\frac{n^k}{(2k)^{2k}}\right),$$

for every equi-partition $\varphi : X \to Y \cup Z$ and $k > 3$.

The second polynomial (defined in Section 5.3.2) has rank that is away from the full rank by a constant factor in the exponent.

Theorem 9. *Let h be the polynomial defined as $h(X) = \sum_{i\in[3]} w_i \left(\prod_{j\in[\frac{k}{2}]} \sum_{(u,v)\in B_{ij}} x_u x_v\right)$. Then there is a constant $c = \frac{23+20\epsilon}{114}$ for a fixed $\epsilon > 0$ such that for every equi-partition $\varphi : X \to Y \cup Z$, over the rational function field $\mathbb{F}(w_1, w_2, w_3)$ such that,*

$$\mathsf{rank}_{\varphi}(h) \geq \left(\frac{n}{k}\right)^{ck}.$$

In Section 5.4 we prove lower bounds on the size of classes of parameterized multilinear polynomials computing the hard polynomials defined in the previous sections. We first obtain lower bound against ROABPs.

Theorem 10. *A ROABP P computing the polynomial family $f = (f_{n,2k})$ requires size*

$$S = \Omega(n^k/(2k)^{2k}).$$

Theorem 11. *An ROABP computing the family of polynomials h defined in Section 5.3 requires size $n^{\Omega(k)}$.*

The following result is the lower bound against strict-interval ABPs.

Corollary 4. *Any strict-interval ABP computing the polynomial f has size $n^{\Omega(\sqrt{k})}$.*

In Section 5.4.4 we obtain a lower bound against sum of ROPs with restricted ordering. The ordering follows from a graph constructed from the ROF computing a ROP. Bisection of an undirected graph $G = (V, E)$ is a set $S \subseteq V$ such that $|S| = |V|/2$. The size of a bisection S is the number of edges across S and $\overline{S}$, i.e., $|\{(u, v) \mid (u, v) \in E, u \in S, v \notin S\}|$. The following is an immediate consequence of the results preceding it, in Section 5.4.4.

Theorem 14. *Let G be a graph on n vertices such that there is a bisection of G of size $n^{1-\epsilon}$. Suppose $p_1, \ldots, p_s$ be ROFs such that G_{p_i} is a sub-graph of G. Then, if $p = p_1 + \cdots + p_s$ we have $s = (n^{\Omega(k)}/t(k))$, where t is a computable function on k.*

In Section 5.4.2, we prove a separation between parameterized read-once and read-2 oblivious ABPs.

Corollary 3. *There is a parameterized polynomial family computable by polynomial size read-2 ABPs such that any ROABP computing it has size $n^{\Omega(k)}$ where k is the parameter.*

5.3 Construction of high rank polynomials

For any complexity measure $\mu : \mathbb{F}[X] \to \mathbb{R}_{\geq 0}$ for polynomials to be useful, we need a class of polynomials where the measure is "small" and an explicit family of polynomials where the measure is “large”. In this section, we consider the latter task and show construction of two parameterized polynomial families $f = (f_{n,2k})_{k\geq 0}$ and $h = (h_{2n,k})$ such that the rank of the partial derivative matrix is large for almost all partitions.

The first family is computable by a depth four circuit of FPT (i.e., $t(k)n^{O(1)}$ where $t(k) = k^{O(k)}$) size. For any partition φ, $\mathsf{rank}_\varphi(f)$ matches the maximum possible value defined by the upper bound described in Lemma 2, upto a factor that depends only on the parameter.

The second family h is a sum of three ROPs, also computable by a circuit of FPT size. In the case of h, $\mathsf{rank}_\varphi(h)$ attains the maximum possible value upto a constant factor in the exponent. Hence, $\mathsf{rank}_\varphi(h) \geq t_2(k)n^{ck}$ where t_2 is a computable function on k and $c < 1/2$.

5.3.1 A full rank polynomial

We know, by Lemma 2, that for a multilinear polynomial g of degree k in n variables, the maximum possible value of $\mathsf{rank}_\varphi(g)$ over all partitions φ is at most $(k+2)\binom{n/2}{k/2}$. Though it is possible to construct polynomials that achieve this bound under a fixed partition φ, it is not immediately clear if there is a polynomial g computed by small circuits that is full rank under every equi-partition. In the following, we give the description of a multilinear polynomial of degree k that has rank $n^{k/2}/t(k)$ where t is a function that depends only on k. We assume that $2k|n$.

We consider K_{2k}, the complete graph on $2k$ vertices. Suppose $V_1 \cup \cdots \cup V_{2k} = X$ be a partition of the variable set $X = \{x_1, \ldots, x_n\}$ such that $|V_i| = |V_j|$ for $1 \leq i < j \leq 2k$. For convenience let $V_i = \{x_{(i-1)n/2k+1}, \ldots, x_{in/2k}\}$, where we assume a natural ordering among the variables, i.e., $x_j \succeq x_i,\ \forall j \geq i$. We consider the variable set V_i as the label of vertex i of the graph K_{2k} for $1 \leq i \leq 2k$. For each edge (i,j) of K_{2k}, we define a polynomial p_{ij} on the vertex set $V_i \cup V_j$. These *edge polynomials* p_{ij} will be used in the subsequent construction of the polynomial f.

Let $\mathcal{M}$ be the set of all possible perfect matchings on $G = K_{2k}$. We define a parameterized family of polynomial $f = (f_{n,2k})_{n>1,2k|n}$, $f_{n,2k}$ as follows:

$$f(x_1, x_2, \ldots, x_n) = \sum_{M \in \mathcal{M}} \zeta_M \prod_{(i,j)\sim M} (1 + p_{ij}(V_i \cup V_j)),$$

where ζ_M for $M \in \mathcal{M}$ are formal variables. We define the edge polynomial p_{ij} as an n/k-variate quadratic multilinear polynomial, such that,

$$p_{ij}(x_{(i-1)n/2k+1}, \ldots, x_{jn/2k}) = \sum_{k<\ell} \omega_{k,\ell} x_k x_\ell.$$

Here $\omega_{i,j}$, for $1 \leq i < j \leq n/k$, are also formal variables. So the polynomial f is defined on $\mathbb{G}[X]$ where $\mathbb{G}$ is an extension of the field $\mathbb{F}$ containing $\{\omega_{i,j}\} \cup \{\zeta_M \mid M \in \mathcal{M}\}$.

Note that $f_{n,2k}$ is a degree $2k$ polynomial in n variables. When n and k are clear from the context, we use f to denote $f_{n,2k}$. Let $\mathbb{G} = \mathbb{F}(\{\zeta_M \mid M \in \mathcal{M}\} \cup \{\omega_{i,j} \mid 1 \leq i < j \leq n/k\})$, i.e., the rational function field of the polynomial ring $\mathbb{F}[\{\zeta_M \mid M \in \mathcal{M}\} \cup \{\omega_{i,j} \mid 1 \leq i < j \leq n/k\}]$.

We note that by definition, f is multilinear and can be computed by a depth-4 circuit, parameterized by the degree, $2k$. As there are $k^{O(k)}$ perfect matchings in $\mathcal{M}$, and a circuit of size $(kn)^{(O(1))}$ can compute the polynomial $\prod_{(i,j)\in M}(1 + p_{ij})$, the size of the depth-4 circuit computing f is at most $k^{O(k)} n^{O(1)}$, which is FPT with k as the parameter and n as the size of the input.

In the remainder of the section, we argue that the polynomial family f defined above has almost full rank under every partition $\varphi : X \to Y \cup Z$, such that $|Y| = |Z| = |X|/2$. Now, the partition function divides the set of variables X into two equal halves, but it might not divide the individual sets $V_i \cup V_j$, the set of variables on which the edge polynomial p_{ij} is defined, in two equal halves. In that case, we define a new quantity, imbalance, as follows.

Definition 21. Consider an equi-partition function $\varphi : X \to Y \cup Z$. A set $V \subset X$ is said to be ℓ-unbalanced with respect to φ if $\frac{|V|}{2} - |\varphi(V) \cap Z| = \ell = |\varphi(V) \cap Y| - \frac{|V|}{2}$.

It may be noted that ℓ can be a positive or negative accordingly as $|\varphi(V) \cap Y| > |\varphi(V) \cap Z|$ or otherwise. Our first observation is, even if the set $V = V_i \cup V_j$ is ℓ-

unbalanced for $\ell < n/4k$ for all edges (i,j), $\mathsf{rank}_\varphi(p_{ij})$ remains large:

Lemma 12. *If $V_i \cup V_j$ is ℓ-unbalanced with respect to a partition $\varphi : X \to Y \cup Z$, then* $\mathsf{rank}_\varphi(p_{ij}) = \Omega(n/2k - |\ell|)$.

Proof. Without loss of generality, we consider the case $\ell > 0$. As already defined, $V_i \cup V_j = \{x_{(i-1)n/2k+1}, \ldots, x_{jn/2k}\}$ and we denote p_{ij} as p for the rest of the proof. Let us assume, for the ease of calculations, $V_i \cup V_j = \{x_1, \ldots, x_{n/k}\}$. Let φ be such that for all $x_q \in V_i \cup V_j$,

$$\varphi(x_q) = \begin{cases} y_q, & \text{if } q \le n/2k + \ell, \\ z_{q-(n/2k+\ell)} & \text{otherwise.} \end{cases} \tag{5.1}$$

Since p is a quadratic polynomial, the rows of M_{p^φ} are indexed by degree at most one monomials $\emptyset, y_1, \ldots y_{n/2k+\ell}$ and degree two monomials of the form $y_i y_j, 1 \le i < j \le n/2k + \ell$. Similarly, the columns are indexed by $\emptyset, z_1, \ldots, z_{n/2k-\ell}$ and degree two monomials $z_i z_j, 1 \le i < j \le n/2k - \ell$.

We claim that the rows and columns indexed by degree 2 monomials will contribute at most 2 to the rank. This is because all row-indexing monomials $y_i y_j$ will have a non-zero entry $\omega_{i,j}$ only corresponding to the first column indexed by $\emptyset$. Similarly, all column-indexing monomials $z_i z_j$ have one non-zero entry along the first row, indexed by $\emptyset$. The first row and the first column can together contribute a rank of at most 2.

Now, it is required to show that the sub-matrix of M_{p^φ} with rows indexed by $\emptyset, y_1, \ldots y_{n/2k+\ell}$ and columns indexed by $\emptyset, z_1, \ldots, z_{n/2k-\ell}$ has rank $\Omega(n/2k - |\ell|)$.

The (y_i, z_j)th entry of M_{p^φ} contains $\omega_{i,n/2k+j}$. The sub-matrix of M_{p^φ} on rows and columns indexed by degree-1 monomials $\emptyset, y_1, \ldots y_{n/2k+\ell}$ and $\emptyset, z_1, \ldots, z_{n/2k-\ell}$ has dimension $n/2k + \ell$ by $n/2k - \ell$. By suitably substituting the formal variables $\omega_{i,n/2k+j}$ with values from $\mathbb{F}$, we can ensure that the sub-matrix of M_{p^φ} is of full column-rank when ℓ is positive. Thus $\mathsf{rank}_\varphi(p) = \Omega(n/2k - \ell)$. Therefore, over any edge (i,j) in G and any φ, the polynomial p_{ij} has rank $\Omega(n/2k - |\ell|)$. It may be noted that the argument above works even when φ does not satisfy (5.1). This completes the proof. $\square$

Before we proceed with proof of the required lower bound on the rank of f under any partition, we define imbalance on each variable set V_i, denoted by $D(V_i)$. If the imbalance on $V_i \cup V_j$ for an edge (i, j) is ℓ, we want $D(V_i), D(V_j)$ to be such that $\ell = D(V_i) + D(V_j)$.

Definition 22. For a partition $\varphi : X \to Y \cup Z$, the imbalance on a set V_i is defined as

$$D(V_i) \overset{\mathsf{def}}{=} |\varphi(V_i) \cap Y| - \frac{|V_i|}{2}.$$

Let $Y_i = \varphi(V_i) \cap Y,\ Z_i = \varphi(V_i) \cap Z$. We know $\forall i \in [2k],\ |V_i| = \frac{n}{2k}$. So, $D(V_i) = |Y_i| - \frac{n}{4k}$ is the imbalance of φ on V_i.

Under a partition φ, in the extreme cases, all variables in V_i are mapped to Y, or none of them are. Hence, $|Y_i| \in [0, \frac{n}{2k}]$, since $|V_i| = n/2k$. It follows that $D(V_i) \in [\frac{-n}{4k}, \frac{n}{4k}]$.

We are now ready to give the rank bound on the polynomial family f.

Theorem 8. *For the parameterized multilinear polynomial family $f = (f_{n,2k})_{n,k\geq 0}$ such that $f(X) = \sum_{M\in\mathcal{M}} \zeta_M \prod_{(i,j)\sim M}(1 + p_{ij})$, we have,*

$$\mathsf{rank}_{\varphi}(f_{n,2k}) = \Omega\left(\frac{n^k}{(2k)^{2k}}\right),$$

for every equi-partition $\varphi : X \to Y \cup Z$ and $k > 3$.

Proof. Let us fix φ to be an arbitrary equi-partition of X. Note that by the definition of f, it is enough to show that for all equi-partitions φ, there exists an optimal matching N and $f_N = \prod_{(i,j)\in N}(1 + p_{ij})$ such that f_N is of full rank i.e. $\mathsf{rank}_{\varphi}(f_N) = \Omega(\frac{n^k}{(2k)^{2k}})$. Then we set $\zeta_N = 1$ and $\zeta_M = 0$ for all other $M \in \mathcal{M}$, so that f is of almost full rank.

Since f_N is multilinear, it is enough to prove that $\forall (i, j) \in N$, $\mathsf{rank}_{\varphi}(p_{ij}) = \Omega(\frac{n}{k^2})$. This would imply that our optimal perfect matching N is such that all edges (i, j) in N have very low imbalance under φ. Our argument is a construction of the required matching N.

Let us consider an arbitrary matching $M \in \mathcal{M}$. We construct N from M. For that purpose, we need to analyse each edge $e = (i, j)$ in the matching M. Hence, we associate a weight to all edges e with respect to φ such that $\mathsf{wt}(e) = |D(V_i) + D(V_j)|$. The weight of the matching M denoted by $\mathsf{wt}(M)$, is the sum of the weights of the edges in M, i.e., $\mathsf{wt}(M) = \sum_{e\in M} \mathsf{wt}(e)$.

In the following, we give an iterative procedure, that given M, produces a matching N with the required properties. The procedure obtains a new matching of smaller weight than the given matching in each iteration. The crucial observation then is that matchings that are weight optimal with respect to the procedure outlined below indeed have the required property.

We say that a matching N is *good* with respect to φ, if $\forall\ e = (i, j) \in N$, the weight does not exceed a threshold t, i.e. $\mathsf{wt}(e) \leq t = n/2k - n/(2k(k-1))$. Note that if M is good then for every edge $(i, j) \in M$, we have $V_i \cup V_j$ is ℓ-unbalanced for some ℓ with $|\ell| \leq n/2k - n/(2k(k-1))$. Then, by Lemma 12 we have $\mathsf{rank}_\varphi(f_M) \geq (n/(2k(k-1)))^k$. In that case, the matching M is the optimal matching N we desire.

Suppose the matching M is not good. Let $e = (i, j) \in M$ be a *bad edge* such that $\mathsf{wt}(e) > n/2k - n/2k(k-1)$. If there are multiple bad edges, e is chosen such that $\mathsf{wt}(e)$ is the maximum, breaking ties arbitrarily. Note that we can assume that $D(V_i)$ and $D(V_j)$ are of the same sign for $\mathsf{wt}(e)$ to have the highest value. Without loss of generality, assume that both $D(V_i)$ and $D(V_j)$ to be non-negative, i.e., $\mathsf{wt}(e) = D(V_i) + D(V_j)$. Since φ is an equi-partition, we have

$$\begin{aligned}
&\sum_{m\in[2k]} D(V_m) = \sum_{m\in[2k]} \left(|Y_m| - \frac{n}{4k}\right) = \sum_{m\in[2k]} |Y_m| - \frac{n}{2} = 0 \\
\implies &\sum_{m\in[2k]\setminus\{i,j\}} D(V_m) = -\mathsf{wt}(e) \\
\implies &\sum_{e'\in M\setminus\{e\}} \mathrm{sgn}(e')\mathsf{wt}(e') = -\mathsf{wt}(e) < \frac{-n}{2k} + \frac{n}{2k(k-1)},
\end{aligned}$$

where $\mathrm{sgn}(e)$ is ± 1 depending on the sign of $\mathsf{wt}(e)$. By averaging, there is an edge $e_1 \in M$ such that,

$$\mathrm{sgn}(e_1)\mathsf{wt}(e_1) = -\frac{\mathsf{wt}(e)}{(k-1)} < \frac{-n}{2k(k-1)} + \frac{n}{2k(k-1)^2}.$$

Suppose $e_1 = (i_1, j_1)$. The idea is that on swapping the end-points (i, j) of the bad edge e with that of the edge e_1, (i_1, j_1), we will get a new matching M' with two new edges, all other edges being from M. We claim that this matching M' has lesser total weight $\mathsf{wt}(M')$ than $\mathsf{wt}(M)$. We can repeat this process to reduce weights of bad edges in M' till we obtain a matching N where all edges are good.

For the ease of analysis, let $D(V_i) = a$, $D(V_j) = b$, $D(V_{i_1}) = c$, $D(V_{j_1}) = d$. The new matching is constructed based on the values of a, b, c and d. Since $c + d = \mathsf{wt}(e_1) < 0$, it must be that either both c, d are negative, or any one of them is negative, i.e. $c < 0, d \geq 0$ or $c \geq 0, d < 0$. Here, we discuss both these cases.

Case 1 Suppose $c, d < 0$. Then, $|a + b| + |c + d| > |a + c| + |b + d|$. We replace the edges (i, j) and (i_1, j_1) by $(i, i_1), (j, j_1)$ to get a new matching M'. We have $\mathsf{wt}(M') < \mathsf{wt}(M)$.

Case 2 Either $c \geq 0$ and $d < 0$ or $c < 0$ and $d \geq 0$. Without loss of generality, assume that $c \geq 0$ and $d < 0$. We argue even if c is positive, it is smaller than at least one of the values a, b, thus making swapping end-points of e with e_1 yield a better matching.

We know, the least value d can have is $-n/4k$. Suppose $c > \frac{n}{4k} - \frac{n}{2k(k-1)} + \frac{n}{2k(k-1)^2}$. Then we have $d < \frac{-n}{2k(k-1)} + \frac{n}{2k(k-1)^2} - c < \frac{-n}{4k}$ which is impossible. Therefore, we have $c \leq \frac{n}{4k} - \frac{n}{2k(k-1)} + \frac{k}{2k(k-1)^2}$.

So, if $c > a, b$, then $a + b < 2c \leq \frac{n}{2k} - \frac{n}{k(k-1)} + \frac{n}{k(k-1)^2}$. For $k > 3$, this is a contradiction since $\mathsf{wt}(e) = a + b > \frac{n}{2k} - \frac{n}{2k(k-1)}$, and this lower bound seems to be higher than the upper bound. Hence, we consider the following sub-cases:

Sub-case (i) $a > c$. Then $a + b > c + b$, replace the edges (i, j) and (i_1, j_1) with the edges (i, j_1) and (i_1, j) to get the new matching M'.

Sub-case (ii) $b > c$. Then $a + b > a + c$, replace (i, j) and (i_1, j_1) with the edges (i, i_1) and (j, j_1) to get the new matching M'.

The second case above also implies that $|b + d| \geq 0$ and $|a + c| \geq \frac{n}{2k} - \frac{n}{2k(k-1)} + \frac{k}{2k(k-1)^2}$. We know $|a + b| \leq n/2k$. Therefore, the least decrease in total weight of matching is:

$$\begin{aligned} |a+b| + |c+d| - |a+c| - |b+d| =& \frac{n}{2k} + \left(\frac{n}{2k(k-1)} - \frac{n}{2k(k-1)^2} \right) \\ & - \left(\frac{n}{2k} - \frac{n}{2k(k-1)} + \frac{k}{2k(k-1)^2} \right) - 0 \\ &= \frac{n}{k(k-1)} - \frac{n}{k(k-1)^2} = \frac{2t}{(k-1)}. \end{aligned}$$

For the new matching M' obtained from M as above, we have one of the following properties:

- It has smaller total weight than M, i.e., $\mathsf{wt}(M') < \mathsf{wt}(M)$, or
- If M has a unique maximum weight edge, then the weight of any edge in M' is strictly smaller than that in M, i.e. $\max_{e' \in M'} \mathsf{wt}(e') < \mathsf{wt}(e)$, or
- The number of edges that have maximum weight in M' is strictly smaller than that in M, i.e., $|\{e'' \mid \mathsf{wt}(e'') = \max_{e' \in M'} \mathsf{wt}(e')\}| < |\{e'' \mid \mathsf{wt}(e'') = \max_{e' \in M} \mathsf{wt}(e')\}|$.

Since all of the invariants above are finite, by repeating the above procedure a finite number of times we get a matching $N \in \mathcal{M}$ such that any of the above steps are not applicable. That is, for every $e' \in N$, $\mathsf{wt}(e') \leq n/2k - n/2k(k-1)$. In fact, the largest value of $\mathsf{wt}(e) = n/2k$, and least decrease is $\mathsf{wt}(M) - \mathsf{wt}(M') \geq t/(k-1) = n/2k(k-1) - n/2k(k-1)^2$ by averaging. So, in at most k iterations for each edge, i.e. $O(k^2)$ iterations, we will obtain a matching where all edges have zero imbalance. The matching we need has low imbalance t for every edge, so our algorithm will obtain N from M in $O(k^2)$ iterations.

As required, for every edge $(i, j) \in N$, we have $\mathsf{rank}_\varphi(p_{ij}) = \Omega(n/2k(k-1))$ and $\mathsf{rank}_\varphi(f_N) = \Omega(n^k/(2k)^{2k})$. By the construction of the polynomial and Lemma 1, we have $\mathsf{rank}_\varphi(f) \geq \max_{M \in \mathcal{M}}\{\mathsf{rank}_\varphi(f_M)\} = \Omega(n^k/(2k)^{2k})$. $\square$

5.3.2 A high rank sum of three ROFs

In Kayal *et al.* (2016), Kayal et al. showed that there is a polynomial that can be written as sum of three ROFs such that any ROABP computing it requires exponential size. The lower bound proof in Kayal *et al.* (2016) is based on the construction of a polynomial using three edge disjoint perfect matchings on n vertices.

The construction of this polynomial, as a crucial ingredient, used a 3-regular mildly explicit family of expander graphs defined in Hoory *et al.* (2006). Let $\mathcal{G} = (G(q))_{q>0,\text{ prime}}$ be a family of 3-regular expander graphs where a vertex x in $G(q)$ is connected to $x+1, x-1$ and x^{-1} where all of the operations are modulo q. When q is clear from the context, we denote $G(q)$ by G. Since G is a simple, undirected graph, we represent any edge e in E, the set of edges in G, by $e = \{u, v\}$.

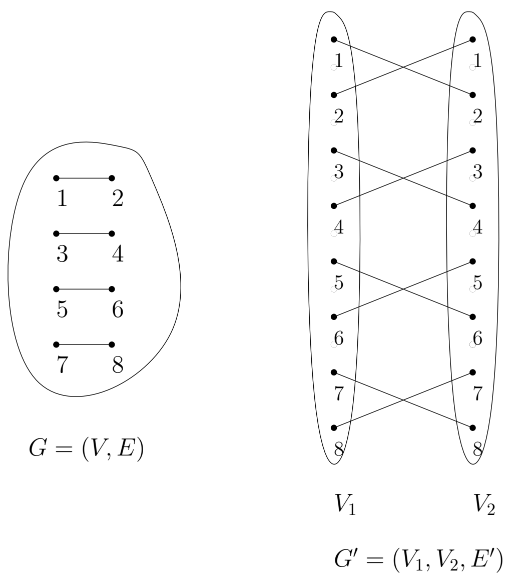


Figure 5.1: Example construction of double cover G' from $G = (V, E)$, where $V = \{1, \dots, 8\}$.

We define G' to be the *double cover* of G such that, $G' = (V_1, V_2, E')$ is the bipartite graph where V_1, V_2 are copies of V and for all pair of vertices $\{u, v\}$ from V such that $u \in V_1, v \in V_2$ and $u \in V_2, v \in V_1$, it holds that $\{u, v\} \in E' \iff \{u, v\} \in E$. As no vertex in G has a self-loop, any edge of the form $\{u, u\}$ is not present in E, and hence, E'.

Figure 5.1 gives an example of a double cover G' of a graph $G = (V, E)$ where $V = \{1, \dots, 8\}$ and $E = (1, 2), (3, 4), (5, 6), (7, 8)$.

It is known from Hoory *et al.* (2006) that the set of edges in E' can be viewed as the union of 3 edge disjoint perfect matchings. In Kayal *et al.* (2016), Kayal et al. construct a polynomial for each of these matchings and the hard polynomial is obtained by taking the sum of these three polynomials. This polynomial has degree $n/2$ and therefore is unsuitable in the parameterized context.

We construct a polynomial h from the same graph G' also based on three edge-disjoint perfect matchings as in Kayal *et al.* (2016), but our polynomial has degree k. Suppose $M_1 \cup M_2 \cup M_3 = E'$ be the edge-disjoint perfect matchings. We divide the $n/2$ edges in each of the M_i into $k/2$ bags of n/k edges each.

Suppose $M_i = B_{i1} \cup B_{i2} \cup \dots \cup B_{ik/2}$, where B_{ij} is the jth bag of edges in M_i. The division of edges into these $k/2$ bags is done arbitrarily. Now, for each edge $(i, j) \in M$, we consider a monomial $x_i x_j$. So, for every bag B_{ij}, we define the bilinear term

$h_{ij} = \sum_{(u,v)\in B_{ij}} x_u x_v$ and the polynomial h_i corresponding to each M_i is defined as $h_i = \prod_{j\in[\frac{k}{2}]} h_{ij}$.

The final polynomial is the following:

$$h(x_1,\ldots,x_n) = \sum_{i\in[3]} w_i \prod_{j\in\frac{k}{2}} h_{ij} = \sum_{i\in[3]} w_i \left(\prod_{j\in[\frac{k}{2}]} \sum_{(u,v)\in B_{ij}} x_u x_v \right), \tag{5.2}$$

where w_1, w_2 and w_3 are formal variables, w_i corresponding to the matching M_i. To analyse the hardness of this polynomial, we need the notion of bichromatic edges.

Definition 23. For a partition $\varphi : X \to Y \cup Z$, and an edge $(u,v) \in M_i$, (u,v) is said to be *bichromatic* with respect to φ if either $\varphi(x_u) \in Y$ and $\varphi(x_v) \in Z$, or $\varphi(x_u) \in Z$ and $\varphi(x_v) \in Y$.

For a set of edges A, let $\mathsf{be}_\varphi(A)$ be the number edges in A that are bichromatic with respect to φ. For a graph $G = (V,E)$, let $\mathsf{be}_\varphi(G) = \mathsf{be}_\varphi(E)$.

Let X be the variable set corresponding to vertices in G, i.e., $X = \{x_1,\ldots,x_n\}$. Fixing an equi-partition φ, we will view it as a coloring of variables in X in the color represented by Y or Z. The degree-2 monomials in the polynomial can now be viewed as a monochromatic edge if both end-points are of the same color, and a bichromatic edge otherwise. As seen in Equation 5.2, in a particular term h_{ij}, each bichromatic edge contributes 1 towards the rank of the partial derivative matrix of h_{ij} and all monochromatic edges together contribute a maximum of 2. This is the idea we will use to prove the desired property of the polynomial h in the following theorem.

Theorem 9. *Let h be the polynomial defined in Equation 5.2. Then there is a constant $c > \frac{23+20\epsilon}{114}$ for a fixed $\epsilon > 0$ such that for every equi-partition $\varphi : X \to Y \cup Z$, over the rational function field $\mathbb{F}(w_1,w_2,w_3)$, we have*

$$\mathsf{rank}_\varphi(h) \geq \left(\frac{n}{k}\right)^{ck}.$$

Proof. Let us fix an arbitrary partition $\varphi : X \to Y \cup Z$. By the expander property of G (see Kayal *et al.* (2016)), the number of edges from Y to Z is lower bounded by $E(Y,Z) \geq \frac{(2+10^{-4})}{2} \cdot |Y| = \frac{(1+\epsilon)n}{2}$ for a fixed $\epsilon > 0$.(See Kayal *et al.* (2016) for details.)

Now, each perfect matching has $\frac{n}{2}$ edges, so the graph has $\frac{3n}{2}$ edges. By averaging, we get that there is a matching M_i, $1 \leq i \leq 3$ such that the number of bichromatic edges in M_i,

$$\mathsf{be}_{\varphi}(M_i) \geq \frac{(1+\epsilon)n}{6}. \tag{5.3}$$

Without loss of generality, suppose $i = 1$. Let $h_1 = \prod_{j \in [\frac{k}{2}]} \sum_{(u,v) \in B_{1j}} x_u x_v$, i.e., the polynomial corresponding to M_1. We need to get an upper bound for $\mathsf{be}_{\varphi}(M_1)$.

Let us assume that the bichromatic edges in M_1 are distributed evenly across all sets in the partition $B_{11}, \ldots, B_{1k/2}$. Then, for every bag B_{1j} we will have $\mathsf{be}_{\varphi}(B_{1j}) = \left(\frac{(1+\epsilon)n}{6} \cdot \frac{2}{k}\right) = \frac{(1+\epsilon)n}{3k}$, which will be the same as $\mathsf{rank}_{\varphi}(h_{1j})$ as each bichromatic edge contributes 1 towards the rank. By sub-multiplicativity, it follows that $\mathsf{rank}_{\varphi}(h_1) \geq \left(\frac{(1+\epsilon)n}{3k}\right)^{k/2}$.

However, this may not hold in general for M_1, because the bichromatic edges can potentially be distributed in exponential number of ways across the bags B_{1j}. Therefore, it is possible that for some values of j, the bags B_{1j} contain only monochromatic edges. This will reduce the exponent of n in the expression of the rank. Thus, this argument is not the right way to analyse the rank of the polynomial.

Nevertheless, we get a smaller but good enough bound by a simple averaging argument. Let $\mathsf{be}_{\varphi}(M_i) = \sum_{j \in [\frac{k}{2}]} \mathsf{be}_{\varphi}(B_{ij})$ and let α denote the number of bags with sufficient number of bichromatic edges, i.e. $\alpha = |\{j \mid \mathsf{be}_{\varphi}(B_{1,j}) \geq n/20k\}|$. Then, for $(k/2 - \alpha)$ bags of M_1, the maximum number of bichromatic edges is upper bounded by $(k/2 - \alpha)n/20k$, and the upper bound for each of the α remaining bags is the total number of edges in each bag, n/k. So, we upper bound $\mathsf{be}_{\varphi}(M_1)$ as follows:

$$\begin{aligned} \mathsf{be}_{\varphi}(M_1) &\leq \alpha \frac{n}{k} + (k/2 - \alpha)\frac{n}{20k} \\ \implies \mathsf{be}_{\varphi}(M_1) &\leq \alpha \frac{n}{k} \cdot \frac{19}{20} + \frac{n}{40}. \end{aligned} \tag{5.4}$$

Finally, using the lower bound given by (5.3) with the upper bound in (5.4), we have

$$\begin{aligned} \frac{(1+\epsilon)n}{6} &\leq \alpha \frac{n}{k} \cdot \frac{19}{20} + k \cdot \frac{n}{40k} \\ \implies \alpha &\geq \frac{(23 + 20\epsilon)}{114} k. \end{aligned}$$

Now $\mathsf{rank}_\varphi(\sum_{(u,v)\in B_{1j}} x_u x_v) = \mathsf{be}_\varphi(B_{1j})$ as we have already explained. Hence we have $\mathsf{rank}_\varphi(h_1) \geq (\frac{n}{20k})^\alpha = \left(\frac{n}{k}\right)^{ck}$ for some constant $c > \frac{(23+20\epsilon)}{114}$ as required. On setting $w_1 = 1$ and w_2, w_3 to zero, we obtain the same rank lower bound for h.

□

5.4 Lower bounds

In this section we prove parameterized lower bounds for some special classes of syntactic multilinear ABPs computing the polynomial families defined in Section 5.3. In particular, we prove lower bounds for the size of ROABPs, strict interval ABPs and a sum of restricted class of ROPs for computing one or both of the hard polynomials we have defined in the previous section.

The general strategy of the lower bound is to obtain an equi-partition φ of the variables for which any polynomial computed by our chosen model attains the largest rank and then comparing this upper bound, which is usually in terms of the size s of the ABP, with the rank lower bound on the hard polynomials.

5.4.1 ROABP

In this section we prove a parameterized lower bound for the size of any ROABP computing the polynomials defined in Section 5.3. The lower bound argument follows from the fact that for any polynomial computed by an ROABP P, there exists an equi-partition φ of variables such that $\mathsf{rank}_\varphi(P)$ is bounded by the size of the ROABP Nisan (1991).

Lemma 13. *A ROABP P of size S computing a polynomial in $\mathbb{F}[X]$ has $\mathsf{rank}_\varphi(P) \leq S$ for an equi-partition $\varphi : X \to Y \cup Z$.*

Proof. Let P be an ROABP of size S computing a polynomial f'. Let the layers in P be $L_0, L_1, \ldots, L_n$, such that L_0 contains only the source node s and L_n contains only the terminal node t. We consider the order in which the variables are read from left to right in the ROABP as $x_1, x_2, \ldots, x_n$.

We can define the equi-partition $\varphi : X \to Y \cup Z$ given the above order, such that,

$$\varphi(x_i) = \begin{cases} y_i, & \text{if } i \leq n/2, \\ z_{i-n/2} & \text{otherwise.} \end{cases}$$

Let us consider the $n/2$th layer. By definition of P, the incoming edges to any layer L_i in P are labelled with a linear polynomial in x_i. At any node j in $L_{n/2}$, the paths from s to j are products of linear terms in variables v, $\varphi(v) \in Y$. The sum of these paths, which is computed at j, can be seen as a sub-program $[s, v_j]_P$. Similarly, the sum of the paths from j to t, computed at t, can be denoted by the sub-program $[v_j, t]_P$. Then, we can represent f as

$$f(x_1, \ldots, x_n) = \sum_{j \in L_{n/2}} [s, v_j]_P \cdot [v_j, t]_P.$$

By definition of φ, for all $v_j \in L_{n/2}$, every product $[s, v_j]_P \cdot [v_j, t]_P$ contributes 1 towards the rank of P. This is because every row-indexing monomial in M_{P^φ} corresponding to paths from s to j has non-zero entries corresponding to the same column-indexing monomials, corresponding to paths from j to t. So by Gaussian elimination, the product of sub-programs $[s, v_j]_P$ and $[v_j, t]$ contribute at most 1 to $\mathsf{rank}_\varphi(P)$. The number of such products in the expression for f is at most $|L_{n/2}|$, the number of nodes in the $(n/2)^{\text{th}}$ layer.

Thus, $\mathsf{rank}_\varphi(P) \leq |L_{n/2}| \leq S$, S being a loose upper bound on the number of nodes in the $n/2$th layer. $\square$

Theorem 10. *A ROABP P computing the polynomial family $f = (f_{n,2k})$ requires size*

$$S = \Omega(n^k/(2k)^{2k}).$$

Proof. By Lemma 13 and given that the polynomial computed by P is f, $\mathsf{rank}_\varphi(f) \leq S$. By Theorem 8, $\mathsf{rank}_\varphi(f) = \Omega(n^k/(2k)^{2k})$. Therefore we have $S = \Omega(n^k/(2k)^{2k})$ as required. $\square$

Combining Lemma 13 with Theorem 9 we get:

Theorem 11. *An ROABP computing the family of polynomials h defined in Equation 5.2 requires size $n^{\Omega(k)}$.*

Proof. From the proof of Theorem 10, we see that for any size S ROABP computing the polynomial h, the equi-partition φ that maps the first $n/2$ variables to Y and the rest to Z ensures that $\mathsf{rank}_{\varphi}(h) \leq S$. Then by Theorem 9, we have $S = n^{\Omega(k)}$ as required. $\square$

5.4.2 Separation Between Read-2 and Read-Once Oblivious ABPs

In this section, we show the power of reads of a variable in an oblivious ABP. To be clear, our family of hard polynomials h can be computed efficiently by oblivious ABPs if we allow two reads instead of one for every variable.

Theorem 12. *The family of polynomials h defined in Section 5.3 can be computed by read-2 oblivious ABPs of size $n^{O(1)}$.*

Proof. According to the construction of h, we have three edge-disjoint perfect matchings M_1, M_2 and M_3. We will try to construct an ABP computing the polynomial f with as less number of reads as possible for each variable. We show that two reads per variable is enough.

We first consider the two perfect matchings M_1 and M_2 in the graph $G' = (V_1, V_2, E')$. By definition of E', for every perfect matching M in G' there will exist a perfect matching N in G, G' being the double cover of G. Hence, corresponding to M_1, M_2 we have edge-disjoint perfect matchings N_1, N_2 in G. As N_1 and N_2 are edge disjoint, their union forms a cycle cover of the graph G, $\mathcal{C} = (C_1, \ldots, C_\ell)$. An arbitrary cycle $C_i = (p_i, x_{i,1}, \ldots, x_{i,r}, p_i)$ for $r > 0$, where the first variable repeats at the end. We call this repeating variable the pivot for the cycle C_i. We construct a read-2 oblivious ABP P computing h_1, h_2 as defined in Section 5.3 and we show that we can add edges to P to compute h_3 without having to increase the number of layers in which a variable is read more than 2.

Given the cycle cover $\mathcal{C}$, we want to construct a read-2 oblivious ABP from it. Our procedure constructs one ROABP for each matching N_1 and N_2, but these ROABPs are on different orders, so their sum is a read-2 oblivious ABP. Each of these ROABPs $P_1,\ P_2$ is a sum of ROABPs P_{ij} on the same order, one corresponding to each disjoint

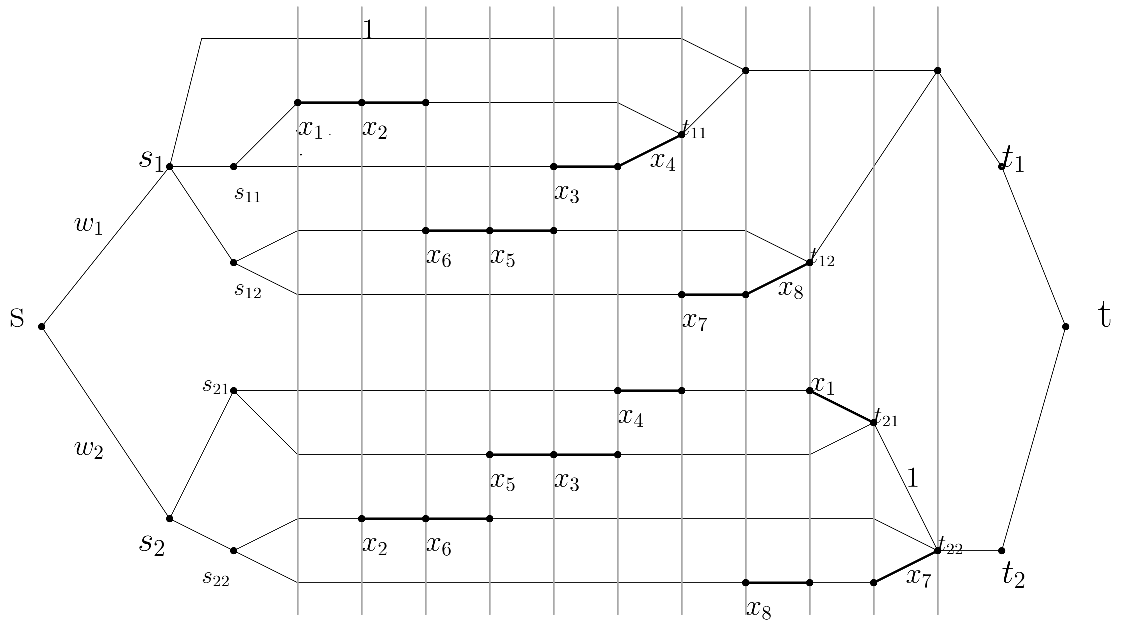


Figure 5.2: Example construction of read-2 oblivious ABP P', where $V = \{1, 2, \ldots, 8\}$ and perfect matchings $N_1 = \{(1, 2), (3, 4), (5, 6), (7, 8)\}$ and $N_2 = \{(1, 4), (2, 6), (3, 5), (7, 8)\}$.

variable set B_{ij} i.e., the ROABP P_1 for N_1 is the sum of ROABPs P_{1j} for B_{1j}s, which start and end at terminal nodes $s_{1j},\ t_{1j}$ (refer to Figure 5.2 for more clarity on the notation).

Each path in P_{ij} corresponds to a monomial $x_m x_n$ in the bilinear term corresponding to $B_i j$. The two orders in which variables will occur in $P_1 + P_2$ is fixed by the cycle cover C. We know, given $\mathcal{C}$, each monomial in $P_1 + P_2$ is now of the form $p_i x_{i,1}$, $x_{i,j}x_{i,j+1}$ and $p_i x_{i,r},\ i \in [\ell]$. We consider the cycles in the order $C_1, \ldots, C_\ell$. The monomial $p_1 x_{11}$ is converted into a path in P_{ij} such that $p_1, x_{11} \in B_{ij}$. This process is repeated as our procedure reads through the edges in the cycles in the cycle cover. This requires taking one edge from the cycle and searching each bag $B_{ij},\ i \in \{1, 2\}$ for it, which will take $2 \times \frac{n}{2k} \cdot \frac{k}{2} = \frac{n}{2}$ for each edge. There are at most n edges in the cycle, so the total time taken (and number of paths added to the ABP) is $O(n^2)$.

We note that the variable p_i is being read twice, while every other variable is read only once. Hence, the pivot variables constitute the second pass on variables and two orders on the variables emerge from a sequential reading of the cycles $C_1, \ldots, C_\ell$.

Now, we consider N_3. For all monomials of the form $p_i p_j$, $x_{i,r}p_i$, where p_i are pivots in the cycle cover, we add paths to the ABP according to the first or second occurrence of the pivot variables. For the rest of the monomials $x_{i,a}x_{j,b}$, we can add them according to their only occurrences in $P_1 + P_2$ or according to another order as we please, as we can allow one more read for the non-pivot variables. We add $n/2$ paths to the previous ABP in total.

Hence, we have a read-2 oblivious ABP P of size $n^{O(1)}$ computing h.

Figure 5.2 illustrates the construction of the read-2 oblivious ABP P' for N_1, N_2 i.e., $w_1h_1 + w_2h_2$ where the set of vertices $V = \{1, 2, \dots, 8\}$ and the perfect matchings are $N_1 = \{(1, 2), (3, 4), (5, 6), (7, 8)\}$ and $N_2 = \{(1, 4), (2, 6), (3, 5), (7, 8)\}$. Hence, they form a cycle cover C_1, C_2 where $C_1 = (1, 2, 6, 5, 3, 4, 1)$, $C_2 = (7, 8, 7)$. Thus x_1, x_7 are pivot variables and are read twice, while the other variables are read once. □

From the above theorem, the separation between read-once and read-twice oblivious ABP is clear.

Corollary 3. *There is a parameterized polynomial family computable by polynomial size read-2 ABPs such that any ROABP computing it has size $n^{\Omega(k)}$ where k is the parameter.*

Proof. Follows from Theorems 11 and 12. □

5.4.3 Strict interval ABPs

In this section we prove a parameterized lower bound against the polynomial family f defined in Section 5.3 for the size of strict interval ABPs. The equivalence of ROABP and strict interval ABP obtained in Section 3.5 already gives us a lower bound against strict interval ABPs. Nevertheless, we describe a lower bound on strict interval ABPs here, as the argument is interesting in its own right and might be generalizable to a larger class of ABPs.

Recalling the definition of strict-interval ABPs (Definition 4), we assume without loss of generality, that π is the identity permutation. Let P be a π strict-interval ABP computing the polynomial f.

As a crucial ingredient in the lower bound proof, we show that using the standard divide and conquer approach, a strict-interval ABP can be transformed into a depth four circuit with $n^{O(\sqrt{k})}$ blow up in the size. To begin with, we need a simple depth reduction for strict interval ABPs computing degree k polynomials. For that purpose, we first homogenize the strict-interval ABP:

Lemma 14. *Let P be a syntactic multilinear ABP of size S computing a homogeneous degree k polynomial g on n variables. Then there is a syntactic multilinear ABP P' of depth $k+1$ and size $O(S \cdot k)$ computing g such that:*

1. *Every node in the i^{th} layer of P' computes a homogeneous degree i polynomial.*
2. *If P is strict interval then so is P'.*

Proof. Without loss of generality, we assume that P is homogeneous, i.e., for every node v in P, the polynomial $[s,v]_P$ is homogeneous, since homogenization of an ABP, first illustrated by Nisan (1991), does not blow up the size of the ABP beyond a factor of the degree k. For every node v in P, let $\deg(v)$ be the degree of the polynomial $[s,v]_P$. We give a layer by layer construction of the program P'. Let L_i be the set of all nodes v in P such that $\deg(v) = i$, i.e., $L_i = \{v \mid \deg(v) = i\}$. The program P' has $k+1$ layers in addition to s and t, with i^{th} layer consisting of nodes L_i. The edges are added inductively as follows:

Base Case: Every node in L_0 computes a degree 0 polynomial, i.e., a constant. Add suitable edges from s to nodes in L_0.

Inductive Step: Suppose the branching program has been constructed upto layer L_{i-1} for $i \geq 1$. We add incoming edges to L_i as follows. For every node $v \in L_i$, with an incoming edge (u,v) in P, we have two possibilities,

Case 1 $u \in L_{i-1}$. Then, we add the edge (u,v) to P' with the same label as $\mathsf{label}(u,v)$.

Case 2 $u \in L_i$. In this case, we wait till all the incoming edges of u are processed. Then, for every incoming edge (u',u) in P' we add the edge (u',v) with the label $\mathsf{label}(u',u) \cdot \mathsf{label}(u,v)$.

There is a one-to-one correspondence between the nodes of P and that of P', since we have only moved nodes of P of degree $< i$ from L_i to a suitable layer preceding L_i, ensuring that nodes j in L_i (in P') compute only sub-programs $[s, v_j]_P$ of degree exactly i.

As the polynomial is of degree k, every node in P will result in the creation of at most k new nodes. Hence P' computes the same polynomial as P and the properties 1 and 2 hold as required. $\square$

Using Lemma 14 we obtain the desired parameterized version of depth reduction to depth four circuits:

Lemma 15. *Let $g(x_1, \ldots, x_n)$ be a multilinear polynomial of degree k computed by a syntactic multilinear branching program P of size S. Then*

$$g(x_1, \ldots, x_n) = \sum_{i=1}^{T} \prod_{j=1}^{\sqrt{k}} f_{i,j} \tag{5.5}$$

for some $T = S^{O(\sqrt{k})}$ and $f_{i,j}$ is a degree $\sqrt{k}$ multilinear polynomial computed by a sub-program of P for $i \in \{1, \ldots, T\}$, $j \in \{1, \ldots, \sqrt{k}\}$.

Proof. The construction follows from a simple divide and conquer subdivision of the program. By Lemma 14, we assume that P is homogeneous and has depth $k+1$.

Let L_i be the set of nodes at layer i of the program P, for $0 \leq i \leq k+1$. We divide P into blocks of $\sqrt{k}$ layers. Each block is a collection of sub-programs between the nodes in the first and last of the $\sqrt{k}$ layers. Let $W = L_{\sqrt{k}} \times L_{2\sqrt{k}} \times \cdots \times L_{(\sqrt{k}-1)\sqrt{k}}$ be the total number of ways in which all these blocks can be aligned with each other. The final polynomial is sum, over all possible alignments, of the product of a sequence of $\sqrt{k}$ sub-programs, one from each block:

$$g(x_1, \ldots, x_n) = \sum_{(i_1, \ldots, i_{\sqrt{k}-1}) \in W} [s, i_1]_P \cdot \prod_{m=1}^{\sqrt{k}-2} [i_m, i_{m+1}]_P \cdot [i_{\sqrt{k}-1}, t]_P \tag{5.6}$$

Every sub-program is a sum of products of linear polynomials. Hence, looking at the expression, we note that it can be computed by a depth-4 circuit. Considering the number of nodes in each layer to be upper bounded by S, we have $|W| = T = S^{O(\sqrt{k})}$. Thus we are able to expand g as in the statement of the Lemma. $\square$

Now to prove the claimed lower bound for the size of strict interval ABPs, all we need is given a polynomial f computed by an strict interval ABP of size S, an equi-partition φ of X such that $\mathsf{rank}_\varphi(f) << n^k$.

Lemma 16. *Let f be a polynomial computed by a strict interval ABP of size S. Then there is a partition φ such that* $\mathsf{rank}_{\varphi}(f) \leq S^{O(\sqrt{k})} n^{\sqrt{k}}$.

Proof. Without loss of generality, assume that P is a strict interval ABP with respect to the identity permutation. Let $\varphi_{\mathsf{mid}} : X \to Y \cup Z$ be a suitable equi-partition such that,

$$\varphi_{\mathsf{mid}}(x_i) = \begin{cases} y_i, & \text{if } i \leq n/2, \\ z_{i-n/2} & \text{otherwise.} \end{cases}$$

We consider the representation for f as in (5.6). Then for every $1 \leq i \leq T$, for all but one m, we have either $\varphi_{\mathsf{mid}}(\mathsf{var}([i_m, i_{m+1}])) \subseteq Y$ or $\varphi_{\mathsf{mid}}(\mathsf{var}([i_m, i_{m+1}])) \subseteq Z$. Now, considering (5.5), $\mathsf{rank}_{\varphi}(f_{ij}) \leq n$ as there are n possibilities for a variable in $\mathsf{var}([i_m, i_{m+1}]_P)$ which is mapped to a different partition by φ_{mid} than the other variables, such that the sub-program $[i_m, i_{m+1}]_P$ contributes a rank of 1.

Therefore, $\mathsf{rank}_{\varphi_{\mathsf{mid}}}([s, i_1]_P \cdot \prod_{m=1}^{\sqrt{k}-2} [i_m, i_{m+1}]_P \cdot [i_{\sqrt{k}-1}, t]_P) \leq n^{\sqrt{k}}$, for every $i_j \in L_{j\sqrt{k}}$.

By sub-additivity of rank_{φ}, we have $\mathsf{rank}_{\varphi}(f) \leq S^{O(\sqrt{k})} n^{\sqrt{k}}$ for $\varphi = \varphi_{\mathsf{mid}}$. □

The required lower bound is immediate now.

Corollary 4. *Any strict-interval ABP computing the polynomial f has size $n^{\Omega(\sqrt{k})}$.*

Proof. Follows from Theorem 8 and Lemma 16. □

5.4.4 Rank bound for ROPs by Graph representation

The reader might be tempted to believe that the lower bound arguments in the preceding sections might be applicable to more general models such as sum of ROFs and sum of ROABPs or even multilinear formulas. However, as we have seen in Section 5.3, there is a sum of three ROFs that has high rank under every partition. Thus our approach using rank_{φ} as a complexity measure is unlikely to yield lower bounds for even sum of ROFs, which is in contrast to the classical setting, where exponential lower bounds against models such as sum of ROFs and sum of ROABPs follow easily.

In this section, we develop a new method of analysing rank of degree k polynomials computed by ROFs. We look at the order in which variables appear in the in-order traversal of an ROF. Thus we read the variables in the sum of ROFs with a restricted ordering.

Let $p \in \mathbb{F}[X]$ be the polynomial computed by a ROF Φ. We want to construct a graph $G_p = (X, E_p)$ corresponding to p so that $\mathsf{rank}_\varphi(p)$ can be related to certain parameters of the graph. For this, we add edges or a sequence of edges to the graph according to the type of polynomial each gate computes. We define some types of gates in the formula as follows.

Definition 24. (Types of gates in a ROF) Let Φ be a ROF. A gate v in Φ is said to be a *maximal-degree-two gate* if v computes a degree two polynomial, and the parent of v computes a polynomial whose degree is strictly greater than two.

A gate v is said to be a *maximal-degree-one* gate if v computes a linear form and the parent of v computes a polynomial of degree strictly greater than one.

A gate v at depth 1 is said to be a *high degree gate* if the degree of the polynomial computed at v is strictly greater than two.

Let V_2 denote the set of all maximal-degree-two gates in Φ, V_1 denote the set of all maximal-degree-one gates and V_0 denote the set of all high degree gates in Φ at depth one. Let $\mathsf{atomic}(\Phi) = V_0 \cup V_1 \cup V_2$. The following is a straightforward observation:

Observation 5. Let Φ be an ROF and v be a maximal-degree-two gate in Φ. Then the polynomial computed by Φ_v is of the form $\Phi_v = \sum_{i=1}^{s} \ell_{i_1}\ell_{i_2}$, where ℓ_{i_j} $1 \leq i \leq s$, $j \in \{1, 2\}$ are variable disjoint linear forms for some $s > 0$ such that each of the ℓ_{i_j} is dependent on at least one variable.

Defining paths and constructing G_p

For a linear form $\ell = \sum_{j=1}^{r} \alpha_{i_j} x_{i_j}$, let $\mathsf{path}(\ell)$ be the simple undirected path comprised of edges (x_{i_1}, x_{i_2}), $(x_{i_2}, x_{i_3}), \ldots, (x_{i_{r-1}}, x_{i_r})$.

In the case when $r = 1$, $\mathsf{path}(\ell)$ is just single vertex. Similarly, for a subset $S \subseteq X$ of variables, let $\mathsf{path}(S)$ denote the path constituted by the edges (x_{i_1}, x_{i_2}), $(x_{i_2}, x_{i_3}), \ldots, (x_{i_{r-1}}, x_{i_r})$ where $S = \{x_{i_1}, \ldots, x_{i_r}\}$, $i_1 < i_2 < \ldots < i_r$.

For two variable disjoint linear forms ℓ and ℓ', let $\mathsf{path}(\ell, \ell')$ be the path obtained by connecting the last vertex in $\mathsf{path}(\ell)$ to the first vertex of $\mathsf{path}(\ell')$ by a new edge.

Now, we define a graph $G_p = (X, E_p)$ where vertices correspond to variables $x_u \in X$ and the set of edges E_p defined as follows.

For each $v \in \mathsf{atomic}(\Phi)$ we add the following edges to E_p:

Case 1 $\Phi_v = \sum_{i=1}^{r} \ell_{i_1}\ell_{i_2}$ for some $r > 0$ add $\mathsf{path}(\ell_{i_1}, \ell_{i_2})$ to G_p for every $1 \leq i \leq t$.

Case 2 $\Phi_v = \prod_{i \in S} x_i$ or $\Phi_v = \sum_{i \in S} c_i x_i$, where $S \subseteq X$, c_is are constants from $\mathbb{F}$, add $\mathsf{path}(S)$ to G_p.

It may be noted that the graph G_p is not unique as it depends on the given minimal ROF Φ computing f. Now that we have an underlying graph, we view the equi-partition φ on the variables X of the polynomial as a coloring, and analyse the rank of the polynomial computed by Φ using the measure of number of bichromatic edges, as defined in Section 5.3.

Lower bound using G_p

In the following, we show that for a given partition φ, we bound the $\mathsf{rank}_\varphi(p)$ in terms of the number of bichromatic edges $\mathsf{be}_\varphi(G_p)$.

Theorem 13. *Let $p \in \mathbb{F}[x_1, \ldots, x_n]$ be a multilinear polynomial of degree k computed by a ROF Φ. Then, for any equi-partition $\varphi : X \to Y \cup Z$,* $\mathsf{rank}_\varphi(p) \leq (4\mathsf{be}_\varphi(G_p))^{\frac{k}{2}}$.

Proof. The proof is by induction on the structure of Φ. The base case is when the root gate of Φ is in $\mathsf{atomic}(\Phi)$. We prove the bound for all the three kinds of gates in $\mathsf{atomic}(\Phi)$ and the inductive argument follows.

Consider a gate $v \in \mathsf{atomic}(\Phi)$.

Case 1 $\Phi_v = \sum_{(i,j) \in S} x_i x_j$. If $\varphi(x_i)$, $\varphi(x_j)$ are not in the same partition, then each monomial $x_i x_j$ contributes 1 towards $\mathsf{rank}_\varphi(p)$. At the same time, the edge (x_i, x_j) added to E_p is bichromatic, so each monomial contributes 1 towards the measure $\mathsf{be}_\varphi(G_p)$ as well.

Case 2 $\Phi_v = \sum_{(a,b)\in T} \ell_a \ell_b$. If, for some $x_i, x_j \in \mathsf{var}(\ell_a)$, $\varphi(x_i), \varphi(x_j)$ are in different partitions, then the linear form ℓ_a contributes 2 towards $\mathsf{rank}_\varphi(\ell_a)$. If the same holds true for ℓ_b, then $\ell_a \ell_b$ would together contribute 4 towards $\mathsf{rank}_\varphi(p)$ and ≥ 2 towards the measure $\mathsf{be}_\varphi(G_p)$.

Case 3 $\Phi_v = \sum_{i\in W_1} c_i x_i$ or $\Phi_v = \prod_{i\in W_2} x_i$ for some $W_1, W_2 \subseteq X$. The first case has been considered already. For the second case, if $\exists x_a, x_b \in W_2$ such that $\varphi(x_a), \varphi(x_b)$ are in different partitions, the polynomial computed by the gate v will contribute a 1 towards $\mathsf{rank}_\varphi(p)$ and at least 1 towards $\mathsf{be}_\varphi(G_p)$, otherwise it contributes 0 towards both measures.

Thus we have verified that the statement is true when the root gate v of Φ is contained in $\mathsf{atomic}(\Phi)$. Suppose $p = p_1 \text{ op } p_2$ for $\text{op} \in \{+, \times\}$ where p_1 and p_2 are variable disjoint and are computed by ROFs. By induction hypothesis, $\mathsf{rank}_\varphi(p_j) \leq (4\mathsf{be}_\varphi(G_{p_j}))^{\frac{k_j}{2}}$ where $k_j = \deg(f_j)$. As $\mathsf{be}_\varphi(G_p) = \mathsf{be}_\varphi(G_{p_1}) + \mathsf{be}_\varphi(G_{p_2})$ and $k = k_1 + k_2$ ($\text{op} = \times$) or $k = \max\{k_1, k_2\}$ ($\text{op} = +$) we have, $\mathsf{rank}_\varphi(f) \leq (4\mathsf{be}_\varphi(G_p))^{\frac{k}{2}}$ as required. □

Recall that the bisection of an undirected graph $G = (V, E)$ is a set $S \subseteq V$ such that $|S| = |V|/2$. The size of a bisection S is the number of edges across S and $\overline{S}$, i.e., $|\{(u, v) \mid (u, v) \in E, u \in S, v \notin S\}|$. The following is an immediate corollary to Theorem 13:

Theorem 14. *Let G be a graph on n vertices such that there is a bisection of G of size $n^{1-\epsilon}$. Suppose $p_1, \ldots, p_s$ be ROFs such that G_{p_i} is a sub-graph of G. Then, if $p = p_1 + \cdots + p_s$ we have $s = (n^{\Omega(k)}/t(k))$, where t is a computable function on k.*

Proof. Let $C = (S, \overline{S})$ be the bisecting cut and $\mathsf{size}(C)$ denote the number of edges across the cut. We fix an equi-partition $\varphi : X \to Y \cup Z$ as follows:

$$\varphi(x_i) \in \begin{cases} Y, & \text{if } i \in S, \\ Z, & \text{otherwise.} \end{cases}$$

Then by Theorem 13, $\mathsf{rank}_\varphi(p_i) \leq (4\mathsf{be}_\varphi(G_{p_i}))^{\frac{k}{2}}$. Since G_{p_i} is a sub-graph of G, we have $\mathsf{be}_\varphi(G_p)) \leq \mathsf{size}(C) \leq n^{1-\epsilon}$. Therefore, $\mathsf{rank}_\varphi(p_i) \leq O_k(n^{(1-\epsilon)k/2})$. By sub-additivity, we have $\mathsf{rank}_\varphi(f) \leq sO_k(n^{(1-\epsilon)k/2})$ where O_k is upto a factor that depends only on a function of k. By Theorem 8, we get $s = \Omega(n^{\epsilon k/2})$. □

5.5 Conclusion

Our results demonstrate the challenges in translating classical arithmetic circuit lower bounds to the parameterized setting, when the degree of the polynomial is the parameter. We get a full rank polynomial that can be computed by depth four arithmetic circuits of FPT size, whereas in the classical setting, full rank polynomials cannot be computed by multilinear formulas of polynomial size Raz (2009).

This makes the task of proving parameterized lower bounds for algebraic computation much more challenging task. Given the application of polynomials, whose degree is bound by a parameter, in the design of efficient parameterized algorithms for many counting problems, we believe that this is a worthy research direction to pursue.

Further, we believe that our results are an indication that study of parameterized complexity of polynomials with degree as the parameter could possibly shed more light on the use of algebraic techniques in parameterized algorithms.

CHAPTER 6

ON A GENERATOR BY SHPILKA AND VOLKOVICH

6.1 Introduction

The problem of *Polynomial Identity Testing* (PIT in short) is, given a polynomial, to test if a polynomial is identically zero i.e., whether a polynomial has no non-zero coefficients for any of its monomials. Naturally, the way in which the input polynomial is represented plays a crucial role in determining the complexity of PIT . The input polynomial f can be represented by an explicit list of coefficients of all possible monomials in n variables. Since the number of all possible monomials in n variables is 2^n, such a list is of exponential in size. To solve the problem of PIT on a polynomial with this representation would involve searching for a non-zero coefficient in this list, which will take linear time in terms of input size, yet in terms of n, the complexity will be exponential ($O(2^n)$). This is an extreme case since the list representation is the largest size representation of a polynomial.

The other extreme is the black-box setting where f is given as an evaluation oracle, i.e., if the algorithm queries the black-box with values $a \in \mathbb{F}^n$, the black-box returns $f(a)$, the value of the polynomial f evaluated at a. Testing identity of a polynomial represented by a black-box is well-studied. A randomized polynomial time algorithm for PIT , where the polynomial is represented by a black-box, has been obtained by Ore (1922), and later DeMillo and Lipton (1978), Schwartz (1980) and Zippel (1979). However, obtaining a deterministic algorithm for black-box PIT remains highly elusive and challenging task. As a partial explanation for the difficulty, Kabanets and Impagliazzo (2004) showed that deterministic sub-exponential algorithms for black-box PIT would imply circuit lower bounds.

There are various other implicit representations of polynomials in the form of arithmetic circuits, algebraic branching programs etc.between the two above-mentioned extremities. In the study of PIT , there are two major representations that are frequently considered, the first being the black-box model under the assumption that polynomial represented by the given black-box has a small circuit representation that is unknown to the algorithm. The second way is to represent the input polynomial by an arithmetic circuit computing it, also known as *white-box* representation.

For the first case of black-box representation, Klivans and Spielman (2001) and later Bläser *et al.* (2009) obtained efficient deterministic algorithm for when the input polynomial had small (polynomial sized) depth-2 circuits (known as i.e., sparse polynomials. Generalizing this to depth-3 circuits turned out to be extremely difficult and efficient algorithms could only be obtained by restricting the top sum gate fan-in (Dvir and Shpilka (2007)) and successive results could only relax the bound on the top fan-in (Kayal and Saraf (2009), Saxena and Seshadhri (2010)). The depth reduction by Agrawal and Vinay (2008) to depth-4 also applies to the case of black-box PIT by giving an efficient black-box PIT for unrestricted arithmetic circuits of polynomial size, given an efficient black-box PIT for a depth-4 circuit of sub-exponential size. This justifies the difficulty of the problem of obtaining efficient black-box PIT for circuits of depth-3 and 4, since such a result can be used to obtain PIT for all polynomials with a small (polynomial sized) arithmetic circuit.

One of the approaches for obtaining deterministic algorithms for black-box PIT is to derandomize the randomized algorithm given by Schwartz (1980) and Zippel (1979). The Schwartz-Zippel algorithm randomly samples a suitably sized subset S of the set of all possible inputs to the polynomial f such that if f is not a zero polynomial, then $f(a)$ is non-zero on a point $a \in S$ with high probability. For this purpose, S must contain sufficient number of elements from the domain $\mathbb{F}^n$ that are not roots of the polynomial f. Such a set S is known as the *hitting set* of the polynomial. Formally, a hitting set for a class of polynomials $\mathcal{C}$ is defined as follows.

Definition 25. A hitting set $\mathcal{H}$ for a class of polynomials $\mathcal{C}$ is a collection of witness assignments $\overline{a} = (a_, \ldots, a_n)$ such that for any polynomial $f \in \mathcal{C}$, $\exists \overline{a} \in \mathcal{H}$ such that $f(\overline{a}) \neq 0$. In other words this collection of assignments *hits* every polynomial in $\mathcal{C}$.

It might be noted that the size of a hitting set $\mathcal{H}$ and the complexity of its construction are crucial in determining the usefulness of $\mathcal{H}$. For example, a hitting set $\mathcal{H}$ that is constructible in time polynomial in n and d (degree of the polynomials in $\mathcal{C}$), would imply a deterministic polynomial time algorithm for black-box PIT of polynomials from the class $\mathcal{C}$. Thus the study of hitting sets for various classes of polynomials is one of the approaches for obtaining efficient deterministic algorithms for PIT .

It is easier to interpret a hitting set as the image set of a family of polynomial maps. Such polynomial maps are referred to as *hitting set generators* in the literature (Shpilka and Yehudayoff (2010)). Formally, a hitting-set generator is defined as follows:

Definition 26. A hitting-set generator is a function $G : \mathbb{F}^t \to \mathbb{F}^n$ that preserves the identity of polynomials in a class $\mathcal{C}$ i.e., $f \not\equiv 0 \iff G(f) \not\equiv 0$. G can be viewed as the n-tuple $(G_1, \ldots, G_n)$, where each $G_i : \mathbb{F}^t \to \mathbb{F}$ has individual degree for each of its variables at most n. Therefore, $G(f)$ has degree bounded by nd where $d = \mathsf{degree}(f)$.

Any hitting-set generator has two parameters, n and t. The question of testing identity of a n-variate polynomial $f \in \mathcal{C}$ boils down to testing the identity of $G(f)$, a polynomial on t variables. Here t is typically a function of n, such that $t = \delta(n) \ll n$.

The study of design of hitting set generators for obtaining PIT has received wide attention. Many hitting set generators defined in the literature have been based on the Kroenecker substitution, which converts any n-variate polynomial $f \in \mathbb{F}[X]$ to a univariate polynomial by substituting x_i with x^{2^i}. We can then use the derandomized Schwartz-Zippel algorithm on the resulting univariate polynomial, which takes polynomial time. An example of Kroenecker substitution being used for testing identity is that of a sparse polynomial identity testing algorithm given by Klivans and Spielman (2001). However, it is clear that this results in exponential blow-up in the degree of the resulting polynomial, hence this technique cannot be directly extended to classes bigger than that of sparse polynomials.

Agrawal *et al.* (2015) defined a hitting-set generator using a controlled Kroenecker-like substitution for ROABPs. Subsequent PIT results in Gurjar *et al.* (2017*a*), Gurjar *et al.* (2017*b*) used variations of the substitution in Agrawal *et al.* (2015) to construct hitting set generators for constant-width ROABPs and sum of constant-many ROABPs respectively.

There have been many constructions of hitting set generators without the use of Kroenecker substitutions. For example, Dvir and Shpilka (2007) constructed a hitting-set generator for bounded top fan-in depth-3 circuits using locally-decodable codes. Shpilka and Volkovich (2009) defined a hitting-set generator and used it to obtain deterministic quasi-polynomial time black-box PIT algorithm for the sum of k pre-processed read-once formulas. Consequently, Minahan and Volkovich (2018) improved this result to a polynomial-time algorithm.

In this chapter we study the Shpilka-Volkovich generator and the results by Minahan and Volkovich using the generator. We now formally define the Shpilka-Volkovich generator as follows.

Definition 27 (Shpilka and Volkovich (2009))**.** Let n be the number of variables and $a_1, a_2, \ldots, a_n$ be distinct elements in the field $\mathbb{F}$. Let $G^i_{n,k} \in \mathbb{F}[y_1, y_2 \ldots y_k, z_1, z_2 \ldots z_k]$ be the polynomial defined as follows:

$$G^i_{n,k}(y_1, y_2 \ldots y_k, z_1, z_2 \ldots z_k) = \sum_{j=1}^{k} L_i(y_j)z_j, \text{ where } L_i(x) = \frac{\prod_{j\neq i}(x - a_j)}{\prod_{j\neq i}(a_i - a_j)},$$

$L_i(x)$ are Lagrangian Interpolation polynomials, and $L_i(a_j) = 1$ if $i = j$. The generator $G_{n,k}$ is defined as $G_{n,k} \triangleq (G^1_{n,k}, \ldots G^n_{n,k})$.

However, when n is clear from the context, we denote $G_{n,k}$ by G_k. So, in the case of Shpilka Volkovich generator (SV-generator in short) , $t = 2k$.

Since the time of its design, the SV-generator has received wide attention in the literature. Anderson *et al.* (2015) obtained quasi-polynomial time algorithm for ACIT (i.e., white-box PIT) on multilinear read-k formulas using the SV-generators as one of the ingredients. Minahan and Volkovich (2018) use the SV-generator for obtaining a complete black-box deterministic algorithm for reconstructions of read once polynomials, while Jansen *et al.* (2010) use it to obtain quasi-polynomial time deterministic PIT for occur-once branching programs, a model defined by the authors as a restricted version of ROABP where each variable is a label to only one edge.

In the parameterized setting, Chauhan and Rao (2015) modified the Schwartz-Zippel algorithm (Schwartz (1980); Zippel (1979)) to obtain a randomized FPT-time parameterized PIT algorithm that uses at most $g(k)\log n$ random bits, where the parameter k is the syntactic degree of the polynomial. The primary component in the algorithm by Chauhan and Rao (2015) is the hitting set generator defined by Shpilka and Volkovich (2009). The main observation in Chauhan and Rao (2015) is as follows:

Proposition 6. (Chauhan and Rao (2015)) Let $f \in \mathbb{F}[X]$ be a polynomial of degree k.

Then $f \equiv 0 \iff G_k(f) \equiv 0$, where G_k is the SV-generator.

In Ghosal *et al.* (2017), we give an ACIT algorithm for depth-3 circuits parameterized by the degree of the polynomial. This is done by using SV-generator on depth-3 circuits to yield a small sum of product of univariate polynomials, for which PIT is known due to Saxena (2008).

The applications of SV-generator for obtaining deterministic algorithms for PIT on special classes of circuits leads to the following question: Can the existing hitting set generators be used to obtain deterministic parameterized algorithms for PIT for larger classes of circuits? More specifically, we ask: what are the classes of circuits where SV-generator can be used to obtain efficient deterministic PIT algorithms?

In this chapter, we consider the approach of obtaining an efficient deterministic algorithm for PIT on a class $\mathcal{C}$ of circuits by using a hitting set generator G such that for every $f \in \mathcal{C}$, $G(f)$ is in a class of circuits with known deterministic PIT algorithms, akin to the PIT algorithm for depth-3 degree-parameterized circuits given by Ghosal *et al.* (2017). This approach for obtaining PIT using the SV-generator does not work when we consider larger classes of circuits. This is because, for any polynomial f with large rank of the polynomial coefficient matrix, the coefficient matrix for $G(f)$ has large rank with high probability. Our proof exploits the structure of the SV-generator. We also generalize this result for families of hitting set generators that are similar to the SV-generator.

Our result indicates that the classes of circuits that contain polynomials whose polynomial coefficient matrices have full rank under every partition are perhaps the hardest instances for obtaining deterministic algorithms for ACIT.

6.2 Chapter Outline

Our main result is the rank preservation property of SV-generators, which we prove in Section 6.5.

Theorem 19. *Let $f \in \mathbb{F}[x_1, \ldots, x_n]$ be a polynomial of degree $\leq k$. Let $g = G_{2k}(f)$. Then,*

$$\exists \varphi, \mathsf{rank}(M_{f^\varphi}) \geq R \implies \mathbf{Pr}_{\varphi'}[\mathsf{rank}(M_{g^{\varphi'}}) = R] \geq \Omega(1/2^{2k}),$$

where the probability is taken over the uniform distribution over the set of all partitions of the variables in g into two parts of equal size.

We show that the property also applies to generators similar to the SV-generator.

Corollary 6. *Let $H = (H_{n,2t})_{1\leq 2t\leq n}$ be an SV like hitting set generator such that $H_{n,2t}$ is a hitting set generator for degree t polynomials on n variables. Let $f = f(x_1, \dots, x_n)$ be any polynomial of degree $t/2$ and $h = H(f)$. Then,*

$$\exists\varphi, \mathsf{rank}(M_{f^{\varphi}}) \geq R \implies \exists\varphi' \mathsf{rank}(M_{h^{\varphi'}}) \geq R.$$

In Section 6.4 we obtain black-box PIT against two multilinear classes of occur-once formulas with powering gates and clustered read-2 formulas.

Theorem 16. *Let $f \in \mathbb{F}[x_1, \dots, x_n]$ be a polynomial computed by a polynomial-sized occur-once formula with squaring gates F, f is non-constant. Then $G_1(f)$ is also non-constant i.e., G_1 is a hitting-set generator for f.*

Theorem 18. *G_7 is a hitting-set generator for the class of clustered read-2 formulas defined in Definition 31.*

In Section 6.3 we describe the properties of SV-generator that are useful throughout this chapter, especially in obtaining the PIT results in Section 6.4.

6.3 Properties of the Shpilka Volkovich Generator

We recall that the SV-generator is denoted by G_k where $G_k : \mathbb{F}^{2k} \to \mathbb{F}^n$, $k < n$. For a polynomial f, $G_k(f)$ is the image of f under G_k, i.e., $G_k(f) = f(G_k^1, \dots, G_k^n)$. The parameter k is usually the degree of the polynomial f for which G_k is a hitting-set generator, but the generator also works for any other value of $k < n$. The property of SV-generator that follows directly from its definition is as follows.

Lemma 17. *Let $f \in \mathbb{F}[X]$ be a n-variate polynomial of degree d and let $G_k(f) \not\equiv 0$. Then f has a hitting-set of size $(nd)^{O(k)}$.*

Proof. Since f is an n-variate polynomial of degree d, the polynomial $G_k(f)$ has degree nd by Definition 26. Since $G_k(f)$ is a $2k$-variate polynomial, and the image of G_k contains the hitting-set for f, f has a hitting-set of size $(nd)^{O(k)}$. $\square$

The size of the hitting-set of the polynomial $G_k(f)$ is $(nd)^{2k}$ when k is the degree of the original polynomial. Therefore, the size of the hitting-set is polynomial in n if k is a constant, and $d = \text{poly}(n)$. Hence, if G_k, for a constant k, is the hitting-set generator for a class of polynomials $\mathcal{C}$, then we have polynomial time deterministic PIT for $\mathcal{C}$. For example, the authors in Minahan and Volkovich (2018) obtain a characterisation of ROFs after applying the SV-generator on a ROF with $k = 1$, and thus obtain polynomial-time deterministic PIT for ROFs and the sum of constant-many ROFs.

The next property follows as a consequence of the previous lemma:

Lemma 18. *If $G_k,\ k > 0$ is a hitting-set generator for a class of n-variate polynomials $\mathcal{C}$, then $G_\ell,\ n > \ell > k$ is also a hitting-set generator for $\mathcal{C}$.*

Proof. The image of G_ℓ, $Im(G_\ell)$ contains $Im(G_k)$ for $k < \ell$ by Definition 26 and Lemma 17. $\square$

The following property follows directly from the definition of G_k:

Lemma 19. *Let f_1, f_2 be polynomials on disjoint sets of variables and G_k be the SV-generator with the parameter k. Then $G_k(f_1 + f_2) = G_k(f_1) + G_k(f_2)$.*

Proof. The proof follows from Definition 27, since G_k is linear over the monomials in a polynomial i.e., $G_k(p) = \sum_{m\in M} c_m G_k(m)$ where $p = \sum_{m\in M} c_m \cdot m$, M being the set of monomials with non-zero coefficients in p. $\square$

In Section 6.5, we study the relationship of SV-generator with the complexity measure of rank of the coefficient matrix of a polynomial. It is also easy to see that any function that is linear over the monomials of a polynomial and preserves the rank for coefficient matrix for some partition of its variables is a hitting-set generator. We state this formally as follows.

Theorem 15. *Consider a function $H : \mathbb{F}^t \to \mathbb{F}^n$, $H = (H_1, \ldots, H_n)$ where $H_i : \mathbb{F} \to \mathbb{F}^t$, $H(x_i) = H_i$. If the rank of a polynomial $p \in \mathbb{F}[X]$ from a class $\mathcal{C}$ under some* *partition $\varphi : X \to A \cup B$ is R, and there is a $\varphi' : \{w_1, \ldots, w_t\} \to Y \cup Z$ such that the rank of $H(p)$ is $\Omega(R)$, then H is a hitting-set generator for $\mathcal{C}$.*

Proof. The proof is by contradiction. Let us assume H is not a hitting-set generator i.e., $p \not\equiv 0,\ H(p) \equiv 0$.

In that case, under an arbitrary partition φ', $\mathsf{rank}_{\varphi'}(H(p)) = 0$, which violates the given condition that the mapping H preserves rank.

For further clarification, we also consider the case where $p \equiv 0,\ H(p) \not\equiv 0$. Let $p = \sum_{m \in M} c_i m_i$, where M is the set of monomials of p, $\forall i,\ c_i$ are constants from the field. By definition of H, $H(p) = \sum_{m \in M} c_i H(m_i)$.

Since $H(p) \not\equiv 0$, there must exist at least one $i,\ c_i \neq 0$. As the monomials in M are distinct, this implies $p \not\equiv 0$, which is a contradiction.

Hence, H is a hitting-set generator. □

We could show that the converse of this statement is true for the Shpilka-Volkovich generator (Theorem 19) and for other generators that behave similar to the SV-generator (Corollary 6) in Section 6.5.

6.4 Application of SV-generator: PIT for small multi-linear models

The characterisation of ROFs obtained by Minahan and Volkovich (2018) being simple and elegant, it seemed natural to attempt to extend it to algebraic circuit models slightly larger than the class of ROFs. To this end, we first define a modified model of read-once formulas, *occur-once formulas with squaring gates*, which we define in Definition 28. We were able to obtain and use a characterisation of $G_t(f)$ similar to the one given for ROFs by Minahan and Volkovich (2018), for our model. This yields an efficient black-box PIT for occur-once formulas with powering gates. We also define a second model in Definition 31 which reads the input variables at most twice. We, then, obtain an efficient black-box PIT for this model using techniques similar to the PIT given in Minahan and Volkovich (2018).

6.4.1 PIT for Occur-once formulas with powering gates

In this section, we first define the model of occur-once formulas with squaring gates. We later generalise the model to include powering gates which compute any power $p \geq 2$ of their input, and show that our PIT algorithm holds for this generalised model as well.

Definition 28. Let F be a syntactically multilinear formula such that every input variable labels exactly one input gate. If g is a gate with children $g_1,\ g_2$ such that $g = g_1 + g_2$ or $g = g_1 \times g_2$, then the gates g_1, g_2 compute polynomials that are variable disjoint. Assume, without loss of generality, that along any root to leaf path in F, no two consecutive gates are both sum gates or product gates. Let h be a squaring gate in F, such that $h = h_1^2$, h_1 being its input, let h_2 be the ancestor of h. Then, h_1, h_2 are not squaring gates.

We denote such a formula F by the term *occur-once formula with squaring gates.*

We note that since the sum and product gates compute addition or multiplication of input polynomials defined on disjoint sets of variables, a squaring gate h has a unique ancestor h_2. Successive sum gates or product gates along a path can be compressed to a single sum gate or a single product gate respectively, to yield a smaller formula computing the same polynomial. Thus every squaring gate lies between two gates that can be sum or product gates. We note the following statement about this model:

Observation 6. The formula F can be constructed from an underlying read-once formula F' such that squaring gates occur only between consecutive sum and product gates along a root to leaf path in F'.

We now give a black-box PIT algorithm for this model using the SV-generator and its characterisation given by Minahan and Volkovich (2018).

Minahan and Volkovich (2018) give a characterisation of G_1, where G_k is the Shpilka-Volkovich generator, on a monomial m, $G_1(m)$. We need the following definition in order to use the characterisation obtained by them.

Definition 29. Let $a_1, \ldots, a_n \in \mathbb{F}$ be the constants used in Definition 27. Then, for $I \subseteq [n]$, we define the polynomial $\Phi_I(w) = \prod_{i \in I}(w - a_i)$, and $\Phi_\emptyset(w) = 1$.

Using the above polynomial, the required characterisation is as follows:

Proposition 7 (Minahan and Volkovich (2018))**.** If $P \in \mathbb{F}[X]$ is a homogeneous polynomial of degree d, where each variable x_i has individual degree upper-bounded by δ, then there is a polynomial $P'(y)$ of degree at most $\delta(|\mathsf{var}(P)| - 1)$ such that

$$G_1(P) = z^d P'(y) \cdot \Phi_{[n]}^{(d-\delta)}(y) \cdot \Phi_{[n]\setminus\mathsf{var}(P)}^{\delta}(y).$$

In particular, there is a polynomial $P'(y)$ of degree at most $d(|\mathsf{var}(P)| - 1)$ such that

$$G_1(P) = z^d P'(y) \cdot \Phi_{[n]\setminus\mathsf{var}(P)}^{d}(y).$$

The authors use this characterisation in an inductive argument to prove G_1 as the hitting-set generator for pre-processed ROFs i.e., ROFs where every variable x_i is replaced by $T_i(x_i)$, a univariate polynomial in x_i. We use the same approach to obtain deterministic polynomial-time PIT for the occur-once formula with squaring gates model. We show that the result can be extended to occur-once formula with powering gates computing powers larger than 2 of their inputs.

Theorem 16. *Let $f \in \mathbb{F}[x_1, \dots, x_n]$ be a polynomial computed by a polynomial-sized occur-once formula with squaring gates F, f is non-constant. Then $G_1(f)$ is also non-constant i.e., G_1 is a hitting-set generator for f.*

Proof. The proof is by induction on the depth D of the gates in the formula F.

Base Case: $D = 1$. Then f is a sum or product of input variables or the square of an input variable. Hence, $f = \sum_{i=1}^{n} c_i x_i + c_0$, $f = c_1 \prod_{i\in S} x_i + c_0$ for some $S \subseteq [n]$ or $f = c_1 x_1^2 + c_0$. We have $G_1(f) \mid_{y=a_1} = c_1' z_1 + c_0$, $G_1(f) \mid_{y=a_1} = c_1' z^{|S|} + c + 0$ or $G_1(f) \mid_{y=a_1} = c_1 z_1^2 + c_0$, all of which are non-constant polynomials.

Inductive Step: For the inductive hypothesis, we assume that the statement of the theorem is true for any sub-formula of F of depth at most $(D-1)$. By definition, for a gate f at depth D, we have three cases, $f = f_1^2 + f_2^2$ or $f = f_1^2 f_2^2 + c_0$ where f_1, f_2 are variable disjoint, or $f = f_1^2$.

The inductive step easily follows from the inductive hypothesis for the last two cases. When $f = f_1^2$, G_1 is a hitting-set generator for f if and only if G_1 is a hitting set generator for f_1. This holds true, since, by the inductive hypothesis, G_1 is a hitting-set generator for f_1 and f is non-constant if and only if f_1 is non-constant.

Let $f = f_1^2 f_2^2 + c_0$, where $f_1,\ f_2$ are variable disjoint. By the inductive hypothesis, G_1 is a hitting-set generator for both f_1, f_2, and $G_1(f) = G_1(f_1^2)G_1(f_2^2) + c_0$. Let $f_1,\ f_2$ be non-constant polynomials. Then both $G_1(f_1),\ G_1(f_2)$ are non-constant, and $G_1(f)$ is also non-constant. If at least one of $f_1,\ f_2$ is a constant, say $f_2 = c$ then $f = c' f_1^2 + c_0,\ c' = c^2$, and we have G_1 as the hitting-set generator for f by the same argument as the previous case.

Finally, we consider the case $f = f_1^2 + f_2^2$. We assume $f \neq 0$. We have, for $d = \text{degree}(f)$, some $i,\ 0 \leq i \leq d$, for which we can assume that the ith degree homogeneous component of f, $H_i(f) = H_i(f_1^2) + H_i(f_2^2) \not\equiv 0$.

Applying G_1 on $H_i(f)$, we get:

$$G_1(H_i(f)) = \sum_{j=0}^{i} G_1(H_j(f_1))G_1(H_{i-j}(f_1)) + \sum_{j=0}^{i} G_1(H_j(f_2))G_1(H_{i-j}(f_2)).$$

Using Theorem 7, we expand each term of the form $G_1(H_a(f)),\ a > 0$ on the right-hand side as follows:

$$\begin{aligned}
G_1(H_i(f)) &= \sum_{j=0}^{i} (z^j P_j^{(1)}(y)\Phi^i_{[n]\setminus V_1})(z^{i-j} P_{i-j}^{(1)}(y)\Phi^i_{[n]\setminus V_1}) \\
&\quad + \sum_{j=0}^{i} (z^j P_j^{(2)}(y)\Phi^i_{[n]\setminus V_2})(z^{i-j} P_{i-j}^{(2)}(y)\Phi^i_{[n]\setminus V_2}) \\
&= z^i \left(\Phi^{2i}_{[n]\setminus V_1} \sum_{j=0}^{i} P_j^{(1)}(y)P_{i-j}^{(1)}(y) + \Phi^{2i}_{[n]\setminus V_2} \sum_{j=0}^{i} P_j^{(2)}(y)P_{i-j}^{(2)}(y) \right) \\
&= \Phi^{2i}_{[n]\setminus V_1 \cup V_2} z^i \left(\Phi^{2i}_{V_2} P^{(1)}(y) + \Phi^{2i}_{V_1} P^{(2)}(y) \right)
\end{aligned}$$

Now, $\Phi^{2i}_{V_2}$ can only be cancelled out by $P^{(2)}(y)$ as the roots of $\Phi^{2i}_{V_1}$ and $\Phi^{2i}_{V_2}$ are all different. But $\deg(P^{(2)}) \leq i \cdot (|V_2| - 1)$ whereas $\deg(\Phi^{2i}_{V_2}) = 2i \cdot |V_2| > i \cdot (|V_2| - 1)$. Hence $\Phi^{2i}_{V_2} P^{(1)}(y) + \Phi^{2i}_{V_1} P^{(2)}(y) \neq 0$, and the coefficient of z^i is non-constant.

Thus, G_1 is a hitting set generator for f. $\square$

This proof can be generalised for a powering gate of any degree $p \geq 2$.

Corollary 5. *Let $f \in \mathbb{F}[x_1, \ldots, x_n]$ be a polynomial computed by a polynomial-sized occur-once formula with powering gates with maximum fan-in p, f is non-constant. Then $G_1(f)$ is also non-constant i.e., a hitting-set generator for f.*

Proof. The inductive argument will follow akin to Theorem 16 for $p > 2$. We have $\Phi_{V_2}^{pi} P^{(1)}(y) + \Phi_{V_1}^{pi} P^{(2)}(y) \neq 0$. Since $\mathsf{deg}(P^{(2)}) \leq i(|V_2| - 1)$ and $\mathsf{deg}(\Phi_{V_2}^{pi}) = pi|V_2| > i(|V_2| - 1)$, $P^{(2)}(y)$ cannot cancel out $\Phi_{V_2}^{pi}$. Hence, we have deterministic PIT for the class of Occur-once formulas with powering gates of fan-in bounded by $p,\ p \geq 2$. $\square$

Since the model of occur-once formula with powering gates is a generalisation of the ROFs using powering gates, we now aim to extend the existing efficient black-box PIT for the sum of ROFs model to a model with higher number of reads of its variables.

6.4.2 PIT for Clustered read-2 formulas

The black-box PIT obtained by Shpilka and Volkovich (2008) for the sum of k ROFs model was by setting the parameter $t = 3k + \log n$ for the SV-generator G_t such that G_t is an efficient hitting-set generator for the sum of k ROFs model, given that $G_t,\ t = \log n$ is an efficient hitting-set generator for the class of ROFs. Later, Minahan and Volkovich (2018) reduced the parameter t to $3k + 1$ for G_t to be an efficient hitting-set generator for sum of k ROFs. We obtain a deterministic PIT for a restricted version of read-2 formulas where the lowest level gate computing a read-twice polynomial is restricted to be a sum of 2 ROFs. Our result uses the same techniques as in Shpilka and Volkovich (2008), Minahan and Volkovich (2018). We first define three types of gates, which will constitute a clustered read-2 formula.

Definition 30. (Types of Gates) Let F be a syntactically multilinear formula. The three types of gates in F are as follows:

Type 1: All product gates g such that $g = g_1 \times g_2 + c$, where c is a a constant and var$(g_1) \cap$ var$(g_2) = \emptyset$, i.e., the input polynomials $g_1,\ g_2$ are variable disjoint.

Type 2: All sum gates h such that $h = h_1 + h_2$, where var$(h_1) \cap$ var$(h_2) = \emptyset$, i.e., the input polynomials $h_1,\ h_2$ are variable disjoint.

Type 3: All sum gates h such that $h = h_1 + h_2$, where var$(h_1) \cap$ var$(h_2) \neq \emptyset$ and h is the only gate of type 3 on all root to leaf paths containing h. Since every gate in a ROF is either of type 1 or of type 2, and all h to leaf paths contain gates of only type 1 or 2, the sub-formula rooted at h is a sum of two ROFs h_1 and h_2.

We formally define our model as follows.

Definition 31. A read-2 syntactically multilinear formula F is called a clustered read-2 formula is every gate in F is of the types 1, 2 or 3 as defined in Definition 30.

Jansen *et al.* (2010) show that $G_{\log n+1}$ also works as a hitting-set generator for occur-once ABPs. The straightforward application of G_1 does not work on OOABPs since we will be substituting each label $a_i x_i + b_i$ with $a_i G_1^i + b_i$ which is a polynomial on the same variables y, z as $a_j G_j + b_j,\ j \neq i,\ a_i, a_j, b_i, b_j$ being constants from the field.

We will now visit the PIT result obtained for our model of clustered read-2 formulas.

The following theorem was proven by Shpilka and Volkovich:

Theorem 17. *If $\mathcal{G}$ is a hitting-set generator for ROFs, then $\mathcal{G} + G_{3k}$ is a hitting-set generator for sum of k-many ROFs.*

By Minahan and Volkovich (2018), G_{3k+1} is seen to be the hitting-set generator for the sum of k-many ROPs. By Definition 31, we substitute $k = 2$ in G_{3k+1} and use other properties of SV-generator to obtain the following polynomial-time deterministic PIT algorithm for clustered read-2 formulas.

Theorem 18. *G_7 is a hitting-set generator for the class of clustered read-2 formulas defined in Definition 31.*

Proof. We know G_1 is a hitting-set generator for ROFs, and G_7 is a hitting-set generator for a sum of two ROFs defined on the same set of variables. By the property of the SV-generator given by Lemma 17, if G_k is a hitting-set generator for a class $\mathcal{C}'$ of polynomials, G_ℓ for any $\ell > k$ is also a hitting-set generator for $\mathcal{C}'$. Hence, G_7 is also a hitting-set generator for ROPs.

Given a clustered read-2 formula F, by Definition 31, we have gates of types 1 (product gate), 2 (sum of variable disjoint polynomials), and 3 (sum of two ROFs). We note that the children of a gate of type 3 are of type 1 or 2 and compute read-once polynomials. Substituting $k = 2$ in the expression for the parameter $t = 3k + 1$ given by Minahan and Volkovich (2018), G_7 is the hitting-set generator for any polynomial computed by a gate in F of type 3.

Let us consider a gate H of type 1 or 2. If the inputs to H are ROPs on disjoint sets of variables, the sub-formula rooted at H is also read-once. By Shpilka and Volkovich (2008), G_1 is a hitting-set generator for the polynomial computed by H. Thus, G_7 is also a hitting-set generator for the ROP computed by the gate H by Lemma 17.

We need to argue G_7 is a hitting-set generator for the polynomial computed by the gate H even when at least one of its inputs is a read-2 polynomial (i.e., at least one of the inputs is a gate H' of type 3, or a gate H'' of type 1 or 2 that lies on a path from root to a gate of type 3). We proceed to prove that G_7 is a hitting-set generator for f_H, the polynomial computed by H.

When H is of type 1 and at least one of its inputs is a read-2 polynomial, $f_H = f_1 f_2$ when at least one of $f_1,\ f_2$ is a read-2 polynomial. Since G_7 is a hitting-set generator for a ROF as well as sum of two ROFs, G_7 is a hitting-set generator for both f_1 and f_2. Since f_1 and f_2 are variable disjoint by definition of H, G_7 is also a hitting-set generator for f_H.

When H is of type 2 and at least one of its inputs is a read-2 polynomial, we assume without loss of generality that one of the inputs of H is a type 3 gate and the other is a gate of type 1 or 2 computing a read-once polynomial. We note that G_7 is a hitting-set generator for both the inputs of H. We need to inspect $G_7(f_H)$ to show that it is indeed a hitting-set generator for $f_H = f_{H_1} + f_{H_2}$, where $f_{H_1},\ f_{H_2}$ are the polynomials computed by the gates H_1 and H_2 respectively.

We generalise the statement and show that if G_K is a hitting-set generator for a class of polynomials $\mathcal{C}$, and f_1, f_2 are non-constant polynomials in $\mathcal{C}$ such that $\mathsf{var}(f_1) \cap \mathsf{var}(f_2) = \emptyset$ and $G_k(f_1), G_k(f_2) \neq 0$, then $G_k(f_1 + f_2) \neq 0$.

We know by linearity of G_k (Lemma 19), $G_k(f_1 + f_2) = G_k(f_1) + G_k(f_2)$. But $G_k(f_1)$ and $G_k(f_2)$ are defined on the same variables $y_1, \ldots, y_k, z_1, \ldots, z_k$, so the monomials in $G_k(f_1)$ and $G_k(f_2)$ might cancel each other to yield a zero polynomial. Hence, G_k might not preserve identity of the polynomial denoted by $f_1 + f_2$. We need to argue that this is not the case.

Let $f_1 = \sum_{m \in \mathcal{M}_h} c_m m$, where $\mathcal{M}_1$ is the set of monomials of f_1 and $\forall m \in \mathcal{M}_1$, c_m is a constant from the field $\mathbb{F}$. We know there is at least one $m \in \mathcal{M}_1$ for which $c_m \neq 0$ since $G_k(f_1) \neq 0 \iff f_1 \neq 0$. Let X_m denote the variables in the support of the monomial m.

We consider f_1', an affine shift of the polynomial f_1 such that $f_1' \equiv h(x_1+a_1, \ldots, x_n + a_n)$. The affine shift of the surviving monomial m, p_m can now be described as follows.

$$\begin{aligned}
p_m &= \prod_{x_j \in X_m} (x_j + a_j) \\
&= \prod_{x_j \in X_m} x_j + \left(\sum_{S \subseteq X_m,\ S = \{x_i\}} a_i \prod_{x_j \in X_m \setminus S} x_j \right) + \ldots \\
&+ \left(\sum_{S \subseteq X_m,\ |S| = \ell} \left(\prod_{x_i \in S} a_i \right) \cdot \left(\prod_{x_j \in X_m \setminus S} x_j \right) \right) + \ldots + a_1 a_2 \ldots a_t.
\end{aligned}$$

By the definition of SV-generator, if we substitute $y_i = \alpha_j$ in $G_k(f_1)$, for all variables $x_\ell, \ell \neq j$, $G_k(x_\ell) = 0$ whereas $G_k(x_j) = z_i$. If a monomial m in f_1 has a non-zero coefficient c_m, then an affine shift of the variables in X is bound to yield a linear term $\ell_j = ux_j + v$ in the expansion of p_m, u, v being constants from $\mathbb{F}$. Thus, in $G_k(p_m)$, only this term survives since all other variables are mapped to zero.

When f_1, f_2 are variable disjoint, $x_j \notin \mathsf{var}(f_2)$. Hence, substituting $y_i = \alpha_j$ in $G_k(f_1 + f_2)$, we will still get $G_k(f_1 + f_2) |_{y_i = \alpha_j} = \ell_j + c$, which is non-constant. Thus, G_k is a hitting-set generator for $f_1 + f_2$.

Proving the argument for the case where H has inputs H_1, H_2, H_1 being a ROF and H_2 being a gate of type 3 implies a proof for the cases where H_2 is a gate of type 1 or 2 computing a read-2 polynomial. This is because the first case proves G_7 is a hitting-set generator for the immediate ancestor of a gate of type 3 in F, and we can recursively apply the above argument for all gates on the root to H path in F to obtain the final result. □

Since we have discussed the properties and some applications of the SV-generator, we want to now understand for which models SV-generator cannot be used to obtain efficient black-box PIT and what inherent property of the generator causes this limitation.

6.5 SV-generator preserves rank

In this section, we show that images of a polynomial f under the SV-generator have many partitions where the coefficient matrix has non-FPT rank provided f has one such partition. More generally, we show that the rank of the coefficient matrix of a polynomial acts as an invariant for the SV-generator.

Theorem 19. *Let $f \in \mathbb{F}[x_1, \ldots, x_n]$ be a polynomial of degree $\leq k$. Let $g = G_{2k}(f)$. Then,*

$$\exists \varphi, \mathsf{rank}(M_{f^\varphi}) \geq R \implies \mathbf{Pr}_{\varphi'}[\mathsf{rank}(M_{g^{\varphi'}}) = R] \geq \Omega(1/2^{2k}),$$

where the probability is taken over the uniform distribution over the set of all partitions of the variables in g into two parts of equal size.

Proof Outline Suppose there is a partition φ of the variables in f such that $\mathsf{rank}(M_{f^\varphi}) \geq R$. In order to prove that rank is preserved under the map G_{2k}, we show that any R linearly independent rows of the M_{f^φ} remain linearly independent in the coefficient matrix of the image polynomial $g = G_{2k}(f)$. However, this does not immediately give a partition in the variables of g so that the coefficient matrix has high rank. We show that, in fact for at least $1/2^{2k}$ fractions of the partitions of variables of g, the coefficient matrix of g has large rank. Figure 6.1 is an illustration of the main idea of this proof that independent rows in the coefficient matrix of the input polynomial f transform into a band of rows in the coefficient matrix for $\hat{G}_{2k}(f)$ under most of the partitions of the variables, where $\hat{G}_{2k}$ is a slight modification on the SV-generator G_{2k}.

Proof. Fix $a_1, \ldots, a_n \in \mathbb{F}$ be distinct elements. Recall that the generator G_{2k} with respect to $a_1, \ldots, a_n$ is defined as $(G_{2k}^1, \ldots, G_{2k}^n)$, i.e., $G_{2k}(x_i) = G_{2k}^i \ \forall i \in \{1, \ldots, n\}$.

Consider :

$$\begin{aligned}
G_{2k}(x_i) &= \sum_{p=1}^{2k} z_p L_i(y_p) \\
&= \sum_{p=1}^{2k} z_p \frac{\prod_{j\neq i}(y_p - a_j)}{\prod_{j\neq i}(a_i - a_j)} \\
&= \sum_{p=1}^{2k} z_p \frac{(y_p - a_1)\dots(y_p - a_{i-1})(y_p - a_{i+1})\dots(y_p - a_n)}{(a_i - a_1)\dots(a_i - a_{i-1})(a_i - a_{i+1})\dots(a_i - a_n)} \\
&= \sum_{p=1}^{2k}\sum_{q=1}^{n} b_p z_p y_p^{n-q}(-1)^q \mathsf{SYM}_{n-1,q-1} \quad \text{(by expanding the product, } b_p \text{ is a constant)} \\
&= \sum_{\substack{p\in[2k] \\ q\in[n]}} z_p y_p^{n-q} c_{pqi} \quad \text{(where } c_{pqi} = b_p(-1)^q \mathsf{SYM}_{n-1,q-1}(a_1,\dots,a_{i-1},a_{i+1},\dots,a_n)).
\end{aligned}$$

Multiplying out any of the k terms obtained above, we get

$$G_{2k}(x_{i_1}x_{i_2}\dots x_{i_k}) = \sum_{\substack{p_1,\dots,p_k\in[2k] \\ q_1\dots q_k\in[n-1]}} z_{p_1}\dots z_{p_k} y_{p_1}^{n-q_1}\dots y_{p_k}^{n-q_k} \prod_{j=1}^{k} c_{p_j q_j i_j}$$

Let $\mathcal{M}_k$ be the set of all degree k monomials in the variables $\{x_1,\dots,x_n\}$, and $\mathcal{S}_{nk}$ be the set of all monomials of the form $\prod_{i\in I} z_i y_i^{n-q_i}$, for all multi-sets $I \subseteq \{1,\dots,2k\}$ of size k and $\mathsf{q} = (q_1,\dots,q_k)$ with $1 \leq q_i \leq n-1$. Let $V = \text{Span}(\mathcal{M}_k)$, and $W = \text{Span}(\mathcal{S}_{nk})$ be the vector spaces spanned respectively by the sets $\mathcal{M}_k$ and $\mathcal{S}_{nk}$. The vector space V contains all polynomials in $\mathbb{F}$ of degree k, and hence the dimension of V is $\binom{n+k}{k}$. Also, dimension of W is bounded by $\binom{4k}{k}n^k$. Note that G_{2k} is a linear map from V to W. Let C be the $\binom{n+k}{k} \times \binom{4k}{k}n^k$ matrix representing G_{2k} as a linear map from V to W. Then, $\forall v \in V,\ G_{2k}(v) = C^T v \in W$. Now, we argue that C has full row-rank.

Claim 2. C has full row-rank.

Proof of claim 2. Suppose C is not of full row rank. Then $\exists\, \alpha_{i_1},\dots,\alpha_{i_r} \in \mathbb{R}$, such that $\sum_{j=1}^{r} \alpha_{i_j} C[i_j] = 0$ with $\alpha_{i_j} \neq 0$ for some j, where $C[i]$ represents the i^{th} row of C, and $r \leq \mathsf{dim}(V)$. Hence, as G_{2k} is linear, we deduce that $\exists v_{i_1},\dots,v_{i_r} \in V$ such that

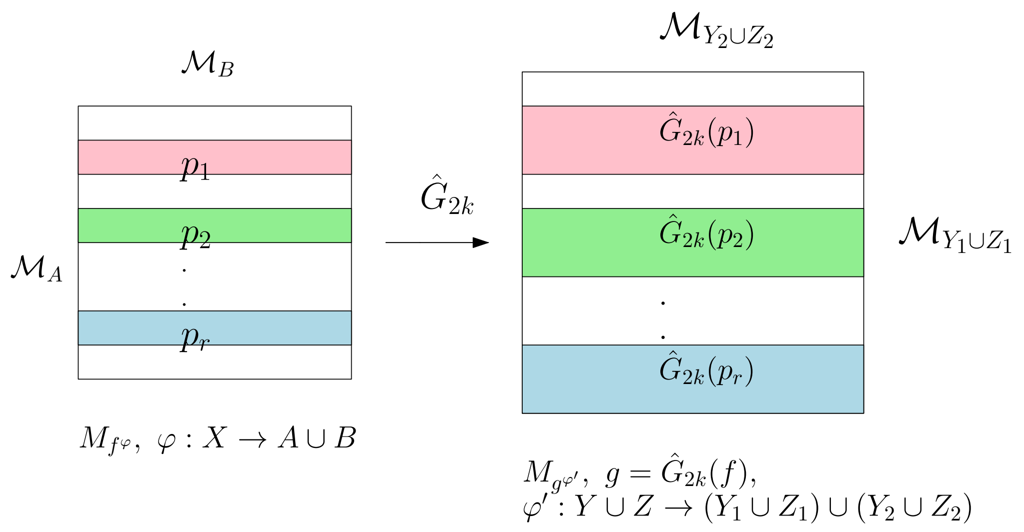

$\mathcal{M}_B$
$\mathcal{M}_A$
p_1
p_2
p_r
$\hat{G}_{2k}$
$\mathcal{M}_{Y_2 \cup Z_2}$
$\mathcal{M}_{Y_1 \cup Z_1}$
$\hat{G}_{2k}(p_1)$
$\hat{G}_{2k}(p_2)$
$\hat{G}_{2k}(p_r)$
$M_{f^\varphi}, \ \varphi : X \to A \cup B$
$M_{g^{\varphi'}}, \ g = \hat{G}_{2k}(f),$
$\varphi' : Y \cup Z \to (Y_1 \cup Z_1) \cup (Y_2 \cup Z_2)$

Figure 6.1: Visualisation of the main proof idea for Theorem 19.

$G_{2k}(v_{i_j}) = C[i_j]$. Then we have:

$$\sum_{j=1}^{r} \alpha_{i_j} G_{2k}(v_{i_j}) = 0 \implies \sum_{j=1}^{r} G_{2k}(\alpha_{i_j} v_{i_j}) = 0 \implies G_{2k}(\alpha_{i_1} v_{i_1} + \ldots + \alpha_{i_r} v_{i_r}) = 0$$

We can see that $P \equiv \alpha_1 v_{i_1} + \ldots + \alpha_{i_r} v_{i_r}$ is a polynomial of degree at most k in $\mathbb{F}[x_1, \ldots, x_n]$, such that $G_{2k}(P) \equiv 0$, whereas $P \not\equiv 0$ since $\exists \alpha_{i_j} \neq 0$. This contradicts Proposition 6. Hence, the claim is proved. $\square$

Consider a partition $\varphi : X \to A \cup B$ and suppose $\mathsf{rank}(M_{f^\varphi}) \geq R$. Let $m_1, \ldots, m_R$ be R linearly independent rows of M_f (chosen arbitrarily). Let $p_1, \ldots, p_R$ be the polynomials representing these rows, i.e., $p_i = \sum_{S \subseteq B} M_f[m_i, m_S] m_S$. Then $p_1, \ldots, p_R$ are linearly independent, i.e., $\forall \alpha_1 \ldots \alpha_R \in \mathbb{F}, \sum_{i=1}^{R} \alpha_i p_i = 0 \implies \forall i, \alpha_i = 0$. Let $q_i = G_{2k}(p_i), 1 \leq i \leq R$ then clearly, $\sum_{i=1}^{R} \alpha_i q_i = 0 \implies \forall i, \alpha_i = 0$.

Suppose $G_{2k} : \mathbb{F}[x_1, \ldots, x_n] \to \mathbb{F}[Y \cup Z]$ where $Y = \{y_1, \ldots, y_{2k}\}$ and $Z = \{z_1, \ldots, z_{2k}\}$. Consider the arbitrary partition: $Y = Y_1 \cup Y_2, \ |Y_1| = |Y_2| = k$. Let $Z = Z_1 \cup Z_2, \ |Z_1| = |Z_2| = k$, where $Z_1 = \{z_i \mid y_i \in Y_1\}$ and $Z_2 = Z \setminus Z_1$. Define the map $\widehat{G_{2k}} = (\widehat{G_{2k}^{(1)}}, \ldots, \widehat{G_{2k}^{(n)}})$, where

$$\widehat{G_{2k}^{(i)}} = \widehat{G_{2k}}(x_i) = \begin{cases} G_{2k}^{(i)}|_{\{w=0 \mid w \in Y_2 \cup Z_2\}} & \text{if } i \in A \\ G_{2k}^{(i)}|_{\{w=0 \mid w \in Y_1 \cup Z_1\}} & \text{if } i \in B \end{cases}$$

Note that the polynomial $G_{2k}^{(i)}|_{\{x=0 \mid x \in Y_2 \cup Z_2\}}$ is indeed a copy of G_k^i for every i, and the same holds for $G_{2k}^{(i)}|_{\{x=0|x\in Y_2\}}$. Hence, $\widehat{G_{2k}}$ is defined over $Y_1 \cup Z_1$ for $i \in A$, and over $Y_2 \cup Z_2$ for $i \in B$. Now, the partition φ naturally induces a partition φ' of $Y \cup Z$.

Let $q_i' = \widehat{G_{2k}}(p_i)$, $m_i' = \widehat{G_{2k}}(m_i)$. Note that $m_1', \ldots, m_R'$ are linearly independent. This follows from the fact that if $\sum_{i\in[R]} \alpha_i m_i' = 0$ and $\exists i \in [R], \alpha_i \neq 0$, then $\sum_{i\in[R]} \alpha_i \widehat{G_{2k}}(m_i) = \widehat{G_{2k}}(\sum_{i\in[R]} \alpha_i m_i) = 0$, since $\widehat{G_{2k}}$ is a linear map. But we know, $\sum_{i\in[R]} \alpha_i m_i \neq 0$ as $m_1, \ldots, m_R$ are distinct monomials. As $\widehat{G_{2k}}$ is a hitting-set generator, $\widehat{G_{2k}}(\sum_{i\in[R]} \alpha_i m_i) \neq 0$.

From the above observations, we have that the polynomials $q_1', \ldots, q_R'$ are linearly independent. Since each of the q_i's correspond to multiple rows (indexed by all possible monomials $Y_1 \cup Z_1$ occurring in q_i') in the matrix $M_{g^{\varphi'}}$, we have $\mathsf{rank}(M_{g^{\varphi'}}) \geq R$. Now, to prove the required probability bound, note that the choice of the partition $Y' = Y_1 \cup Y_2$ was arbitrary, and the choice of the partition $Z' = Z_1 \cup Z_2$ follows from that, since $\forall y_j \in Y_1, z_j \in Z_1$. Hence, the rank bound holds for all the $\binom{2k}{k}$ such partitions of Y. Thus $\mathbf{Pr}[\mathsf{rank}(M_{g^{\varphi'}}) \geq R] \geq \binom{2k}{k}/\binom{4k}{2k} = \Omega(1/2^{2k})$. $\square$

It may be noted that the for $R = n^{\Omega(k)}$ there are degree k polynomials computed by $\Pi\Sigma\Pi$ circuits where there is a partition φ such that Theorem 19 is applicable. Here is an example:

Example The polynomial $p = \prod_{i=0}^{\frac{k}{2}} \left(x_{\frac{in}{2k}+1} x_{\frac{in}{2k}+2} + \ldots + x_{\frac{(i+1)n}{2k}-1} x_{\frac{(i+1)n}{2k}} \right)$ has rank $n^{k/2}/2k^{k/2}$ under the partition φ such that $\forall$ odd $i \in [n], \varphi(x_i) \in Y$, else $\varphi(x_i) \in Z$.

It is not clear if Theorem 19 can be generalized to arbitrary hitting set generators for polynomials parameterized by the degree. The main challenge here is to obtain a partition under which the image of the generator has rank R. The crucial property of the SV-generator that is used to obtain a partition is the fact that substituting a suitable subset of variables to zero results in a a copy of the generator with a fewer number of variables. In fact, it may noted that any family of generators whose suitably chosen projections give a generator from the same family. We call such generators *SV-like* generators.

Let $H = (H_{n,t})_{1\leq t\leq n}$ be a family of generators, where $H_{n,t} : \mathbb{F}[x_1, \ldots, x_n] \to \mathbb{F}[y_1, \ldots, y_t]$. We say that $H_{n,t}$ is a *SV like* generator, if for $1 \leq t \leq n$, $H_{n,t}(x_i)$ there exists a subset $S \subseteq \{y_1, \ldots, y_t\}, |S| = t/2$ such that, $H_{(n,t)} \mid_{\{y=0|y\in S\}}$ and

$H_{(n,t)} \mid_{\{y=0|y \notin S\}}$ are both copies of $H_{(n,t/2)}$.

Corollary 6. *Let $H = (H_{n,2t})_{1 \le 2t \le n}$ be an SV like hitting set generator such that $H_{n,2t}$ is a hitting set generator for degree t polynomials on n variables. Let $f = f(x_1, \ldots, x_n)$ be any polynomial of degree $t/2$ and $h = H(f)$. Then,*

$$\exists \varphi, \mathsf{rank}(M_{f^{\varphi}}) \ge R \implies \exists \varphi' \mathsf{rank}(M_{h^{\varphi'}}) \ge R.$$

Proof. The argument is essentially the same as in Theorem 19. Let $k = t/2$, and let $\mathcal{M}_k$ be the set of all degree k monomials in the variables $\{x_1, \ldots, x_n\}$, and $\mathcal{S}_{n,k}$ be the set of all monomials that appear in at least one of the polynomials in the image set of $\mathcal{M}_k$ under the map $H_{n,2t}$. Note that $\mathcal{S}_{n,k}$ is a finite set. Let $V = \text{Span}(\mathcal{M}_k)$, and $W = \text{Span}(\mathcal{S}_{n,k})$ be the vector spaces spanned by the sets of monomials. The vector space V contains all polynomials in $\mathbb{F}$ of degree k. As in Claim 1, it can be concluded that H is a linear map from V to W that has full row-rank.

Consider a partition $\varphi : X \to A \cup B$ such that $\mathsf{rank}(M_{f^{\varphi}}) \ge R$. Let $m_1, \ldots, m_R$ be row indices of R linearly independent rows of M_f (chosen arbitrarily). Let $p_1, \ldots, p_R$ be the polynomials represented by these rows, i.e., $p_i = \sum_{S \subseteq B} M_f[m_i, m_S] m_S$. Then the polynomials $p_1, \ldots, p_R$ are linearly independent, i.e., $\forall \alpha_1 \ldots \alpha_R \in \mathbb{F}, \sum_{i=1}^{R} \alpha_i p_i = 0 \implies \forall i, \alpha_i = 0$. Let $q_i = G_{2k}(p_i), 1 \le i \le R$ then clearly, $\sum_{i=1}^{R} \alpha_i q_i = 0 \implies \forall i, \alpha_i = 0$. Consider the partition of $Y = \{y_1, \ldots, y_{2t}\}$ to $Y_1 \cup Y_2$, where $Y_1 = S, Y_2 = \{y_1, \ldots, y_{2t}\} \setminus S$. We have $|Y_1| = |Y_2| = t$. Define the map $\widehat{H_{(n,2t)}} = (\widehat{H^{(1)}}, \ldots, \widehat{H^{(n)}})$, where

$$\widehat{H^{(i)}} = \widehat{H}(x_i) = \begin{cases} H^{(i)}_{(n,2t)}|_{\{w=0|w \in Y_2\}} & \text{if } i \in A \\ H^{(i)}_{(n,2t)}|_{\{w=0|w \in Y_1\}} & \text{if } i \in B \end{cases}$$

Note that the polynomial $H^{(i)}_{(n,2t)}|_{\{x=0|x \in Y_2\}}$ is indeed a copy of $H^{i}_{(n,t)}$ for every i, and the same holds for $H^{(i)}_{(n,2t)}|_{\{x=0|x \in Y_1\}}$. Hence, $\widehat{H_{(n,2t)}}$ is defined over Y_1 for $i \in A$, and over Y_2 for $i \in B$. Thus, the partition φ naturally induces a partition φ' of Y.

Let $q_i' = \widehat{H_{(n,2t)}}(p_i)$, $m_i' = \widehat{H_{(n,2t)}}(m_i)$. Now we argue that $m_1', \ldots, m_R'$ are linearly independent, since $m_1, \ldots, m_R$ are linearly independent. Suppose not, and let $\alpha_1, \ldots, \alpha_R \in \mathbb{F}$ be such that $\sum_{i=1}^{R} \alpha_i m_i' = 0$. Since $\widehat{H_{n,2t}}$ is a linear map from V to

W, we have $\widehat{H_{(n,2t)}}(\sum_{i=1}^{R}\alpha_i m_i) = \sum_{i=1}^{R}\alpha_i \widehat{H_{(n,2t)}}(m_i) = \sum_{i=1}^{R}\alpha_i m_i' = 0$, a contradiction to the fact that $\widehat{H_{(n,2t)}}$ is a copy of $H_{n,t}$ and is a hitting set generator for degree k polynomials.

From the above observations, we have that the polynomials $q_1', \ldots, q_R'$ are linearly independent. Since each of the q_i's correspond to multiple rows in the matrix $M_{h^{\varphi'}}$, we have $\mathsf{rank}(M_{h^{\varphi'}}) \geq R$.

$\square$

6.6 Conclusion

We have showed that the SV-generator, with suitable parameters, preserves the rank of partial derivative matrix of a n-variate polynomial with suitable partition of the variables. The main hurdle in generalizing our technique to arbitrary hitting set generators for degree k polynomials is the lack of structure of the generators under substitution of variables. It would be interesting to see if general hitting sets preserve the rank of partial derivative matrix, or rather, any complexity measure.

Finally, it will be interesting to see if hitting set generators can reduce the complexity of a polynomial. More precisely, suppose $\mathcal{C}_1$ and $\mathcal{C}_2$ are algebraic complexity classes such that the class $\mathcal{C}_1$ is subsumed by the class $\mathcal{C}_2$. Let G be a family of hitting set generators for $\mathcal{C}_1$. We ask if it is possible that $G(f) \in \mathcal{C}_2$ for every polynomial f computed by a circuit $C \in \mathcal{C}_1$.

CHAPTER 7

CONCLUSIONS AND FUTURE DIRECTIONS

Syntactically multilinear formulas are a widely studied sub-class of the multilinear circuits. There has been several results obtaining lower bounds on restricted classes of syntactic multilinear formulas computing hard multilinear polynomials, including classes where the number of reads of the input variable is bounded. By varying the number and order of reads allowed in a syntactically multilinear formula or an ABP, it is possible to define novel, interesting models . Obtaining lower bounds and separations between such classes illustrate the power of reads in a circuit. Our results in this thesis further explore this direction of research. In one such result we observe that reading the input variables in intervals in ABPs grant the same power as reading the variables only once, in a fixed order. In order to examine the relationships between classes of bounded-read syntactically multilinear formulas, we construct or use previously defined multilinear polynomials that exhibit the maximum value of the complexity measure of rank of the partial derivative matrix. We proceed to obtain lower bounds against some bounded-read multilinear classes computing such an explicit multilinear polynomial, thus emphasising that the rank measure and the explicit multilinear polynomials continue to be useful in yielding super-polynomial or larger lower bounds on bounded-read multilinear classes.

It is interesting to generalize the problem of obtaining lower bounds further, to models where the degree is represented as a parameter, since it elucidates the power of degree in algebraic computation. In the results in this thesis, we observe that parameterization makes it harder to solve the lower bound problem. Considering the rank of the partial derivative matrix as our complexity measure, we see that the notion of full rank in the bounded-degree setting is easily attained by restricted multilinear parameterized models.

In a different result, we observe that in the parameterized setting, depth four circuits do not wield the same power as unbounded depth circuits as in the classical setting. But the classes of parameterized multilinear circuits of depth four, sum of three ROFs and read-2 oblivious ABPs compute multilinear polynomials of degree k that attain full rank of the partial derivative matrix. Thus it is difficult to obtain separations between parameterized models of computation of larger depth or higher number of reads, respectively, using this complexity measure.

However, since the parameterized paradigm offers a relaxed notion of tractability, it is interesting to inspect if better algorithms for PIT can be obtained on polynomials parameterized by the degree, k. We explore the possibility of obtaining parameterized PIT by using the hitting-set generator defined by Shpilka and Volkovich (2008). The SV-generator is useful in the parameterized setting since it reduces a n-variate polynomial of degree k to a $2k$-variate polynomial of degree nk. Therefore, the SV-generator can be used to obtain white-box PIT for depth three parameterized circuits, since the resulting $O(k)$-variate polynomial takes a simple form for which efficient white-box PIT is known. But the complexity of a depth three circuit, represented by the rank of the partial derivative matrix for any polynomial of degree k computed by a circuit of depth three, is seen to be much lower than that of polynomials computed by circuits of higher depth. For polynomials computed by circuits of depth larger than three, we show that the $O(k)$-variate polynomial yielded by SV-generator has similar value of the rank measure as the original polynomial on which the SV-generator is applied, and does not take a simple form as in the case of depth three parameterized polynomials. We are able to generalize this property to all hitting set generators that are similar to the SV-generator. Thus, in order to obtain parameterized PIT , classes of hitting set generators that resemble the SV-generator will not be useful and more novel approaches are necessary.

Prospective Directions from this Thesis

The most immediately following question from our lower bound results would be to extend the used techniques for obtaining lower bounds against other bounded-read classes like read-k formulas, interval ABPs and interval formulas. Such lower bounds and constructions of explicit multilinear polynomials for the purpose of obtaining these lower bounds may also yield separations between bounded-read multilinear classes. For example, a separation between the classes of syntactically multilinear formulas and sum of read-k formulas (for a constant k) will establish the power of the number of reads in a formula. Similar results can be obtained regarding the power of reading variables in differing orders through obtaining separation between classes of oblivious ABPs and k-pass ABPs.

Regarding parameterized lower bounds, since measures related to the dimension of the space spanned by partial derivatives of a polynomial are by definition close to the measure of rank of partial derivative matrix of a polynomial, it is unlikely that better lower bounds can be obtained using measures like the dimension of shifted partial derivatives. But there is a possibility that other parameterizations of polynomials, or the use of a complexity measure unrelated to the dimension of partial derivatives may yield improved lower bounds and separations between parameterized circuit classes. A parameterized depth reduction of circuits to constant depth might provide more insight on which classes of parameterized polynomials might be more amenable for obtaining lower bounds.

From our result on the rank-preserving property of SV and SV-like generators, it can be concluded that subsequent work on parameterized PIT can make use of other classes of hitting set generators as discussed in this thesis. Such generators might reduce the input polynomial to a different low rank model than a small sum of product of univariate polynomials. One such low rank model can be constant-depth ROABPs, for which black-box PIT is known. Black-box PIT can also be obtained by using generators that reduce the input polynomial to a polynomial on a constant number of variables.

REFERENCES

1. **Agrawal, M.**, **R. Gurjar**, **A. Korwar**, and **N. Saxena** (2015). Hitting-sets for ROABP and sum of set-multilinear circuits. *SIAM J. Comput.*, **44**(3), 669–697. URL `https://doi.org/10.1137/140975103`.

2. **Agrawal, M.** and **V. Vinay**, Arithmetic circuits: A chasm at depth four. *In 49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008, October 25-28, 2008, Philadelphia, PA, USA*. IEEE Computer Society, 2008. URL `https://doi.org/10.1109/FOCS.2008.32`.

3. **Amini, O.**, **F. V. Fomin**, and **S. Saurabh** (2012). Counting subgraphs via homomorphisms. *SIAM J. Discrete Math.*, **26**(2), 695–717. URL `https://doi.org/10.1137/100789403`.

4. **Anderson, M.**, **D. van Melkebeek**, and **I. Volkovich** (2015). Deterministic polynomial identity tests for multilinear bounded-read formulae. *Comput. Complex.*, **24**(4), 695–776. URL `https://doi.org/10.1007/s00037-015-0097-4`.

5. **Arvind, V.** and **S. Raja** (2016). Some lower bound results for set-multilinear arithmetic computations. *Chicago J. Theor. Comput. Sci.*, **2016**. URL `http://cjtcs.cs.uchicago.edu/articles/2016/6/contents.html`.

6. **Baur, W.** and **V. Strassen** (1983). The complexity of partial derivatives. *Theor. Comput. Sci.*, **22**, 317–330. URL `https://doi.org/10.1016/0304-3975(83)90110-X`.

7. **Biedl, T. C.**, **E. D. Demaine**, **C. A. Duncan**, **R. Fleischer**, and **S. G. Kobourov** (2004). Tight bounds on maximal and maximum matchings. *Discret. Math.*, **285**(1-3), 7–15. URL `https://doi.org/10.1016/j.disc.2004.05.003`.

8. **Bläser, M.** and **C. Engels**, Parameterized valiant's classes. *In* **B. M. P. Jansen** and **J. A. Telle** (eds.), *14th International Symposium on Parameterized and Exact Computation, IPEC 2019, September 11-13, 2019, Munich, Germany*, volume 148 of *LIPIcs*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. URL `https://doi.org/10.4230/LIPIcs.IPEC.2019.3`.

9. **Bläser, M.**, **M. Hardt**, **R. J. Lipton**, and **N. K. Vishnoi** (2009). Deterministically testing sparse polynomial identities of unbounded degree. *Inf. Process. Lett.*, **109**(3), 187–192. URL `https://doi.org/10.1016/j.ipl.2008.09.029`.

10. **Brent, R. P.** (1974). The parallel evaluation of general arithmetic expressions. *J. ACM*, **21**(2), 201–206. URL `https://doi.org/10.1145/321812.321815`.

11. **Chauhan, A.** and **B. V. R. Rao**, Parameterized analogues of probabilistic computation. *In* **S. Ganguly** and **R. Krishnamurti** (eds.), *Algorithms and Discrete Applied Mathematics - First International Conference, CALDAM 2015, Kanpur, India, February 8-10, 2015. Proceedings*, volume 8959 of *Lecture Notes in Computer Science*. Springer, 2015. URL `https://doi.org/10.1007/978-3-319-14974-5_18`.

12. **Chung, F. R.**, *Separator theorems and their applications*. Forschungsinst. für Diskrete Mathematik, 1989. URL http://www.math.ucsd.edu/~fan/mypaps/fanpap/117separatorthms.pdf.

13. **DeMillo, R. A.** and **R. J. Lipton** (1978). A probabilistic remark on algebraic program testing. *Inf. Process. Lett.*, **7**(4), 193–195. URL https://doi.org/10.1016/0020-0190(78)90067-4.

14. **Downey, R. G.** and **M. R. Fellows**, *Parameterized Complexity*. Monographs in Computer Science. Springer, 1999. ISBN 978-1-4612-6798-0. URL https://doi.org/10.1007/978-1-4612-0515-9.

15. **Dvir, Z.**, **G. Malod**, **S. Perifel**, and **A. Yehudayoff**, Separating multilinear branching programs and formulas. *In* **H. J. Karloff** and **T. Pitassi** (eds.), *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*. ACM, 2012. URL https://doi.org/10.1145/2213977.2214034.

16. **Dvir, Z.** and **A. Shpilka** (2007). Locally decodable codes with two queries and polynomial identity testing for depth 3 circuits. *SIAM J. Comput.*, **36**(5), 1404–1434. URL https://doi.org/10.1137/05063605X.

17. **Engels, C.** (2016). *Why are certain polynomials hard?: A look at non-commutative, parameterized and homomorphism polynomials*. Ph.D. thesis, Saarland University. URL https://nbn-resolving.org/urn:nbn:de:bsz:291-scidok-64387.

18. **Fischer, I.** (1994). Sums of like powers of multivariate linear forms. *Mathematics Magazine*, **67**(1), 59–61. URL https://doi.org/10.1080/0025570X.1994.11996185.

19. **Flum, J.** and **M. Grohe**, *Parameterized Complexity Theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2006. ISBN 978-3-540-29952-3. URL https://doi.org/10.1007/3-540-29953-X.

20. **Fomin, F. V.**, **D. Lokshtanov**, **V. Raman**, **S. Saurabh**, and **B. V. R. Rao** (2012). Faster algorithms for finding and counting subgraphs. *J. Comput. Syst. Sci.*, **78**(3), 698–706. URL http://dx.doi.org/10.1016/j.jcss.2011.10.001.

21. **Fournier, H.**, **N. Limaye**, **G. Malod**, and **S. Srinivasan**, Lower bounds for depth 4 formulas computing iterated matrix multiplication. *In* **D. B. Shmoys** (ed.), *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*. ACM, 2014. URL https://doi.org/10.1145/2591796.2591824.

22. **Ghosal, P.**, **O. Prakash**, and **B. V. R. Rao**, On constant depth circuits parameterized by degree: Identity testing and depth reduction. *In* **Y. Cao** and **J. Chen** (eds.), *Computing and Combinatorics - 23rd International Conference, COCOON 2017, Hong Kong, China, August 3-5, 2017, Proceedings*, volume 10392 of *Lecture Notes in Computer Science*. Springer, 2017. URL https://doi.org/10.1007/978-3-319-62389-4_21.

23. **Grigoriev, D.** and **M. Karpinski**, An exponential lower bound for depth 3 arithmetic circuits. *In* **J. S. Vitter** (ed.), *Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing, Dallas, Texas, USA, May 23-26, 1998*. ACM, 1998. URL https://doi.org/10.1145/276698.276872.

24. **Grigoriev, D.** and **A. A. Razborov** (2000). Exponential lower bounds for depth 3 arithmetic circuits in algebras of functions over finite fields. *Appl. Algebra Eng. Commun. Comput.*, **10**(6), 465–487. URL https://doi.org/10.1007/s002009900021.

25. **Gupta, A.**, **P. Kamath**, **N. Kayal**, and **R. Saptharishi** (2014). Approaching the chasm at depth four. *J. ACM*, **61**(6), 33:1–33:16. URL https://doi.org/10.1145/2629541.

26. **Gurjar, R.**, **A. Korwar**, and **N. Saxena** (2017*a*). Identity testing for constant-width, and any-order, read-once oblivious arithmetic branching programs. *Theory of Computing*, **13**(1), 1–21. URL https://doi.org/10.4086/toc.2017.v013a002.

27. **Gurjar, R.**, **A. Korwar**, **N. Saxena**, and **T. Thierauf** (2017*b*). Deterministic identity testing for sum of read-once oblivious arithmetic branching programs. *Comput. Complex.*, **26**(4), 835–880. URL https://doi.org/10.1007/s00037-016-0141-z.

28. **Hoory, S.**, **N. Linial**, and **A. Wigderson** (2006). Expander graphs and their applications. *Bulletin of the American Mathematical Society*, **43**(4), 439–561. URL https://doi.org/10.1090/S0273-0979-06-01126-8.

29. **Jansen, M. J.**, Lower bounds for syntactically multilinear algebraic branching programs. *In* **E. Ochmanski** and **J. Tyszkiewicz** (eds.), *Mathematical Foundations of Computer Science 2008, 33rd International Symposium, MFCS 2008, Torun, Poland, August 25-29, 2008, Proceedings*, volume 5162 of *Lecture Notes in Computer Science*. Springer, 2008. URL https://doi.org/10.1007/978-3-540-85238-4_33.

30. **Jansen, M. J.**, **Y. Qiao**, and **J. Sarma**, Deterministic black-box identity testing pi-ordered algebraic branching programs. *In IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2010, December 15-18, 2010, Chennai, India*. 2010. URL https://doi.org/10.4230/LIPIcs.FSTTCS.2010.296.

31. **Jerrum, M.** and **M. Snir** (1982). Some exact complexity results for straight-line computations over semirings. *J. ACM*, **29**(3), 874–897. URL https://doi.org/10.1145/322326.322341.

32. **Kabanets, V.** and **R. Impagliazzo** (2004). Derandomizing polynomial identity tests means proving circuit lower bounds. *Comput. Complex.*, **13**(1-2), 1–46. URL https://doi.org/10.1007/s00037-004-0182-6.

33. **Kayal, N.**, **V. Nair**, and **C. Saha**, Separation between read-once oblivious algebraic branching programs (roabps) and multilinear depth three circuits. *In* **N. Ollinger** and **H. Vollmer** (eds.), *33rd Symposium on Theoretical Aspects of Computer Science, STACS 2016, February 17-20, 2016, Orléans, France*, volume 47 of *LIPIcs*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016. URL https://doi.org/10.4230/LIPIcs.STACS.2016.46.

34. **Kayal, N.**, **C. Saha**, and **R. Saptharishi**, A super-polynomial lower bound for regular arithmetic formulas. *In* **D. B. Shmoys** (ed.), *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*. ACM, 2014. URL https://doi.org/10.1145/2591796.2591847.

35. **Kayal, N.** and **S. Saraf**, Blackbox polynomial identity testing for depth 3 circuits. *In 50th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2009, October 25-27, 2009, Atlanta, Georgia, USA*. IEEE Computer Society, 2009. URL `https://doi.org/10.1109/FOCS.2009.67`.

36. **Kayal, N.** and **N. Saxena** (2007). Polynomial identity testing for depth 3 circuits. *Comput. Complex.*, **16**(2), 115–138. URL `https://doi.org/10.1007/s00037-007-0226-9`.

37. **Klivans, A. R.** and **D. A. Spielman**, Randomness efficient identity testing of multivariate polynomials. *In* **J. S. Vitter**, **P. G. Spirakis**, and **M. Yannakakis** (eds.), *Proceedings on 33rd Annual ACM Symposium on Theory of Computing, July 6-8, 2001, Heraklion, Crete, Greece*. ACM, 2001. URL `https://doi.org/10.1145/380752.380801`.

38. **Koblitz, N.** (1987). Elliptic curve cryptosystems. *Mathematics of computation*, **48**(177), 203–209. URL `https://doi.org/10.1090/S0025-5718-1987-0866109-5`.

39. **Koiran, P.** (2012). Arithmetic circuits: The chasm at depth four gets wider. *Theor. Comput. Sci.*, **448**, 56–65. URL `https://doi.org/10.1016/j.tcs.2012.03.041`.

40. **Koutis, I.**, Faster algebraic algorithms for path and packing problems. *In* **L. Aceto**, **I. Damgård**, **L. A. Goldberg**, **M. M. Halldórsson**, **A. Ingólfsdóttir**, and **I. Walukiewicz** (eds.), *Automata, Languages and Programming, 35th International Colloquium, ICALP 2008, Reykjavik, Iceland, July 7-11, 2008, Proceedings, Part I: Tack A: Algorithms, Automata, Complexity, and Games*, volume 5125 of *Lecture Notes in Computer Science*. Springer, 2008. URL `https://doi.org/10.1007/978-3-540-70575-8_47`.

41. **Koutis, I.** and **R. Williams**, Limits and applications of group algebras for parameterized problems. *In* **S. Albers**, **A. Marchetti-Spaccamela**, **Y. Matias**, **S. E. Nikoletseas**, and **W. Thomas** (eds.), *Automata, Languages and Programming, 36th International Colloquium, ICALP 2009, Rhodes, Greece, July 5-12, 2009, Proceedings, Part I*, volume 5555 of *Lecture Notes in Computer Science*. Springer, 2009. URL `https://doi.org/10.1007/978-3-642-02927-1_54`.

42. **Kumar, M.** and **S. Saraf**, Superpolynomial lower bounds for general homogeneous depth 4 arithmetic circuits. *In* **J. Esparza**, **P. Fraigniaud**, **T. Husfeldt**, and **E. Koutsoupias** (eds.), *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part I*, volume 8572 of *Lecture Notes in Computer Science*. Springer, 2014. URL `https://doi.org/10.1007/978-3-662-43948-7_62`.

43. **Kumar, M.** and **S. Saraf** (2015). The limits of depth reduction for arithmetic formulas: It's all about the top fan-in. *SIAM J. Comput.*, **44**(6), 1601–1625. URL `https://doi.org/10.1137/140999220`.

44. **Mahajan, M.** and **V. Vinay** (1997). Determinant: Combinatorics, algorithms, and complexity. *Chicago J. Theor. Comput. Sci.*, **1997**. URL `http://cjtcs.cs.uchicago.edu/articles/1997/5/contents.html`.

45. **Miller, V. S.**, Use of elliptic curves in cryptography. *In* **H. C. Williams** (ed.), *Advances in Cryptology — CRYPTO '85 Proceedings*. Springer Berlin Heidelberg, Berlin, Heidelberg, 1986. ISBN 978-3-540-39799-1. URL `https://doi.org/10.1007/3-540-39799-X_31`.

46. **Minahan, D.** and **I. Volkovich** (2018). Complete derandomization of identity testing and reconstruction of read-once formulas. *TOCT*, **10**(3), 10:1–10:11. URL `https://doi.org/10.1145/3196836`.

47. **Mitzenmacher, M.** and **E. Upfal**, *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005. ISBN 978-0-521-83540-4. URL `https://doi.org/10.1017/CBO9780511813603`.

48. **Müller, M.** (2008). *Parameterized Randomization*. Ph.D. thesis, Albert-Ludwigs-Universität Freiburg im Breisgau. URL `https://d-nb.info/993356915/34`.

49. **Nisan, N.**, Lower bounds for non-commutative computation (extended abstract). *In* **C. Koutsougeras** and **J. S. Vitter** (eds.), *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing, May 5-8, 1991, New Orleans, Louisiana, USA*. ACM, 1991. URL `https://doi.org/10.1145/103418.103462`.

50. **Nisan, N.** and **A. Wigderson** (1997). Lower bounds on arithmetic circuits via partial derivatives. *Comput. Complex.*, **6**(3), 217–234. URL `https://doi.org/10.1007/BF01294256`.

51. **Ore, Ø.** (1922). über höhere kongruenzen. *Norsk Mat. Forenings Skrifter*, **1**(7), 15.

52. **Ramya, C.** and **B. V. R. Rao**, Lower bounds for special cases of syntactic multilinear abps. *In* **L. Wang** and **D. Zhu** (eds.), *Computing and Combinatorics - 24th International Conference, COCOON 2018, Qing Dao, China, July 2-4, 2018, Proceedings*, volume 10976 of *Lecture Notes in Computer Science*. Springer, 2018. URL `https://doi.org/10.1007/978-3-319-94776-1_58`.

53. **Ramya, C.** and **B. V. R. Rao**, Lower bounds for multilinear order-restricted abps. *In* **P. Rossmanith**, **P. Heggernes**, and **J. Katoen** (eds.), *44th International Symposium on Mathematical Foundations of Computer Science, MFCS 2019, August 26-30, 2019, Aachen, Germany*, volume 138 of *LIPIcs*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019*a*. URL `https://doi.org/10.4230/LIPIcs.MFCS.2019.52`.

54. **Ramya, C.** and **B. V. R. Rao** (2019*b*). Lower bounds for sum and sum of products of read-once formulas. *TOCT*, **11**(2), 10:1–10:27. URL `https://doi.org/10.1145/3313232`.

55. **Raz, R.** (2006). Separation of multilinear circuit and formula size. *Theory of Computing*, **2**(6), 121–135. URL `https://doi.org/10.4086/toc.2006.v002a006`.

56. **Raz, R.** (2009). Multi-linear formulas for permanent and determinant are of super-polynomial size. *J. ACM*, **56**(2), 8:1–8:17. URL `https://doi.org/10.1145/1502793.1502797`.

57. **Raz, R.** and **A. Shpilka** (2005). Deterministic polynomial identity testing in non-commutative models. *Comput. Complex.*, **14**(1), 1–19. URL https://doi.org/10.1007/s00037-005-0188-8.

58. **Raz, R.** and **A. Yehudayoff** (2008). Balancing syntactically multilinear arithmetic circuits. *Comput. Complex.*, **17**(4), 515–535. URL https://doi.org/10.1007/s00037-008-0254-0.

59. **Reed, I. S.** and **G. Solomon** (1960). Polynomial codes over certain finite fields. *Journal of the Society for Industrial and Applied Mathematics*, **8**(2), 300–304. URL https://doi.org/10.1137/0108018.

60. **Saptharishi, R.**, **S. Chillara**, and **M. Kumar** (2016). A survey of lower bounds in arithmetic circuit complexity. Technical report. URL https://github.com/dasarpmar/lowerbounds-survey/releases.

61. **Saxena, N.**, Diagonal circuit identity testing and lower bounds. *In* **L. Aceto**, **I. Damgård**, **L. A. Goldberg**, **M. M. Halldórsson**, **A. Ingólfsdóttir**, and **I. Walukiewicz** (eds.), *Automata, Languages and Programming, 35th International Colloquium, ICALP 2008, Reykjavik, Iceland, July 7-11, 2008, Proceedings, Part I: Tack A: Algorithms, Automata, Complexity, and Games*, volume 5125 of *Lecture Notes in Computer Science*. Springer, 2008. URL https://doi.org/10.1007/978-3-540-70575-8_6.

62. **Saxena, N.** and **C. Seshadhri**, From sylvester-gallai configurations to rank bounds: Improved black-box identity test for depth-3 circuits. *In 51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*. IEEE Computer Society, 2010. URL https://doi.org/10.1109/FOCS.2010.9.

63. **Schwartz, J. T.** (1980). Fast probabilistic algorithms for verification of polynomial identities. *J. ACM*, **27**(4), 701–717. URL https://doi.org/10.1145/322217.322225.

64. **Shpilka, A.** and **I. Volkovich**, Read-once polynomial identity testing. *In* **C. Dwork** (ed.), *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*. ACM, 2008. URL https://doi.org/10.1145/1374376.1374448.

65. **Shpilka, A.** and **I. Volkovich**, Improved polynomial identity testing for read-once formulas. *In* **I. Dinur**, **K. Jansen**, **J. Naor**, and **J. D. P. Rolim** (eds.), *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, 12th International Workshop, APPROX 2009, and 13th International Workshop, RANDOM 2009, Berkeley, CA, USA, August 21-23, 2009. Proceedings*, volume 5687 of *Lecture Notes in Computer Science*. Springer, 2009. URL https://doi.org/10.1007/978-3-642-03685-9_52.

66. **Shpilka, A.** and **I. Volkovich** (2015). Read-once polynomial identity testing. *Comput. Complex.*, **24**(3), 477–532. URL https://doi.org/10.1007/s00037-015-0105-8.

67. **Shpilka, A.** and **A. Yehudayoff** (2010). Arithmetic circuits: A survey of recent results and open questions. *Foundations and Trends® in Theoretical Computer Science*,

5(3—4), 207–388. ISSN 1551-305X. URL http://dx.doi.org/10.1561/0400000039.

68. **Srinivasan, S.** (2019). Strongly exponential separation between monotone VP and monotone VNP. *CoRR*, **abs/1903.01630**. URL http://arxiv.org/abs/1903.01630.

69. **Tavenas, S.** (2015). Improved bounds for reduction to depth 4 and depth 3. *Inf. Comput.*, **240**, 2–11. URL https://doi.org/10.1016/j.ic.2014.09.004.

70. **Valiant, L. G.** (1979). The complexity of computing the permanent. *Theor. Comput. Sci.*, **8**, 189–201. URL https://doi.org/10.1016/0304-3975(79)90044-6.

71. **Valiant, L. G.**, **S. Skyum**, **S. Berkowitz**, and **C. Rackoff** (1983). Fast parallel computation of polynomials using few processors. *SIAM J. Comput.*, **12**(4), 641–644. URL https://doi.org/10.1137/0212043.

72. **Williams, R.** (2009). Finding paths of length k in $o^*(2^k)$ time. *Inf. Process. Lett.*, **109**(6), 315–318. URL https://doi.org/10.1016/j.ipl.2008.11.004.

73. **Yehudayoff, A.**, Separating monotone VP and VNP. *In* **M. Charikar** and **E. Cohen** (eds.), *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*. ACM, 2019. URL https://doi.org/10.1145/3313276.3316311.

74. **Zippel, R.**, Probabilistic algorithms for sparse polynomials. *In* **E. W. Ng** (ed.), *Symbolic and Algebraic Computation, EUROSAM '79, An International Symposiumon Symbolic and Algebraic Computation, Marseille, France, June 1979, Proceedings*, volume 72 of *Lecture Notes in Computer Science*. Springer, 1979. URL https://doi.org/10.1007/3-540-09519-5_73.

LIST OF PAPERS BASED ON THESIS

PUBLICATION IN A REFEREED JOURNAL

1. Purnata Ghosal, B. V. Raghavendra Rao, *A note on parameterized polynomial identity testing using hitting set generators*, *Information Processing Letters*, Volume 151, 2019.

PUBLICATIONS IN CONFERENCE PROCEEDINGS

1. Purnata Ghosal, B. V. Raghavendra Rao, *On Constant Depth Circuits Parameterized by Degree: Identity Testing and Depth Reduction*, *International Computing and Combinatorics Conference (COCOON) 2017*, Hong Kong, August 3-5, 2017.
2. Purnata Ghosal, B. V. Raghavendra Rao, *On Proving Parameterized Size Lower Bounds for Multilinear Algebraic Models*, *International Computing and Combinatorics Conference (COCOON) 2019*, Xian, China, July 29-31, 2019.

MANUSCRIPT UNDER REVIEW

1. Purnata Ghosal, B. V. Raghavendra Rao, *On Proving Parameterized Size Lower Bounds for Multilinear Algebraic Models*. Under review with *Fundamenta Informatica*.

MANUSCRIPT IN PREPARATION

1. Purnata Ghosal, B. V. Raghavendra Rao, *Limitations of Sums of Bounded-Read Formulas*.

CURRICULUM VITAE

NAME:	Purnata Ghosal
DATE OF BIRTH:	2 May,1992
EDUCATIONAL QUALIFICATIONS:	
Bachelor of Engineering	
Institute:	Indian Institute of Engineering Science and Technology, Shibpur
Major:	Computer Science and Technology
Year:	2010-2014

DOCTORAL COMMITTEE

CHAIR PERSON:	Dr. C. Chandrasekhar Professor Department of Computer Science and Engineering
GUIDE:	Dr. B. V. Raghavendra Rao Associate Professor Department of Computer Science and Engineering
MEMBERS	Dr. Jayalal Sarma Associate Professor Department of Computer Science and Engineering
	Dr. Meghana Nasre Assistant Professor Department of Computer Science and Engineering
	Dr. Pradeep Sarvepalli Associate Professor Department of Electrical Engineering